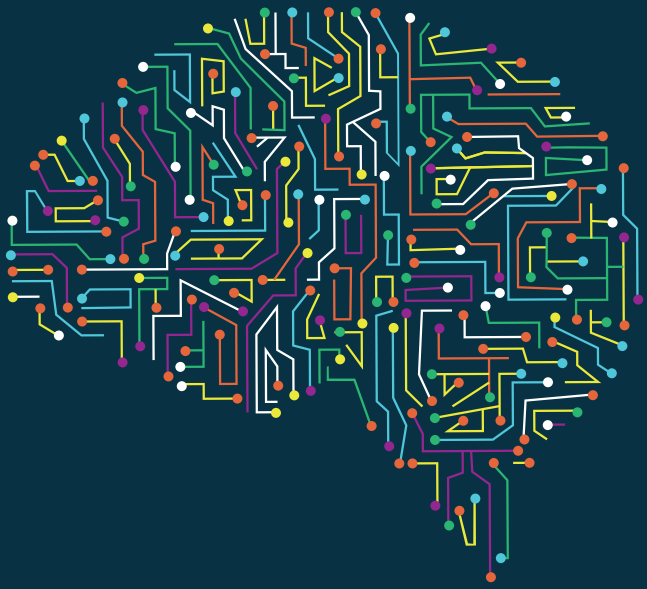


PENSAMIENTO COMPUTACIONAL

UN APORTE PARA LA EDUCACIÓN DE HOY

gurises unidos

Telefónica FUNDACIÓN / movistar

PENSAMIENTO COMPUTACIONAL

UN APORTE PARA LA EDUCACIÓN DE HOY

gurises unidos

Telefónica FUNDACIÓN / movistar

Realización
Esta publicación ha sido elaborada por Gurises Unidos en el marco del proyecto Robotic-Pensamiento Computacional de Fundación Telefónica – Movistar.

Autores Gurises Unidos
Florencia Artecona
Emilio Bonetti
Clara Darino
Federico Mello
Marianela Rosá
Mauro Scópise

Gurises Unidos
Carlos Roxlo 1320, Montevideo, Uruguay.
Tel-fax: (+ 598) 24003081
Web: www.gurisesunidos.org.uy

Coordinación general del Proyecto
Beatriz Ríos
Fundación Telefónica - Movistar
San Martín 2842
www.fundaciontelefonica.uy

Revisión
Ana Clara Pisón
Fundación Telefónica - Movistar

Prólogo – Fundación Telefónica - Movistar

Estamos viviendo una época de profundos cambios; más que una época de cambios se trata de un cambio de época.

Nunca en la historia de la humanidad una generación ha vivido un cambio de una dimensión comparable a la actual, que transformará todas las facetas de la vida. Estamos ante una revolución que afectará a las personas, a las industrias, a todo, de una forma que aún nos es difícil imaginar.

Tendrá implicaciones profundas en el mercado de trabajo y, si no se aborda correctamente, puede dar lugar a desajustes en el ámbito de la economía, la política, y lo social. Telefónica – Movistar ha decidido que quiere centrar gran parte de su labor social en el ámbito de la educación digital porque considera que la educación es un pilar fundamental en la construcción de una nueva sociedad, mejor y más justa.

Vivimos en un mundo en el cual la tecnología es transversal a todos los ámbitos de la vida y nos desafía a desarrollar propuestas innovadoras que no se limitan al trabajo en el aula. Desde Fundación Telefónica – Movistar apostamos por proyectos de educación digital que utilizan la tecnología como soporte.

Hace tres años empezamos a desarrollar el trabajo con Pensamiento Computacional y hoy se ha transformado en un proyecto solvente. El Pensamiento Computacional apunta a generar en los niños una forma de pensar donde aprendan a plantearse problemas y sus soluciones, cumpliendo una secuencia determinada de pasos en el proceso. En este tiempo pudimos corroborar, con el trabajo del equipo en territorio y las diferentes experiencias que se realizan en otros países, que es una propuesta que aporta valor al trabajo en el aula.

Diseñar y aplicar esta metodología fue un desafío que nos obligó a reflexionar sobre distintos aspectos de la tecnología educativa. Con la guía y ayuda del psicólogo Roberto Balaguer empezamos un camino de formación y experimentación en una propuesta nueva para la educación en Uruguay. A través del equipo de trabajo de Gurises Unidos, una de las organizaciones con la que llevamos adelante el proyecto, validamos la metodología que llevábamos a los centros educativos.

Luego de tres años de trabajo incesante, surge esta publicación como forma de sistematizar lo aprendido. Para facilitar su lectura, el libro presenta una introducción sobre el concepto de Pensamiento Computacional y luego una serie de prácticas para llevar este método al aula. Conscientes de que hemos asumido un gran reto, aspiramos a que este material sea de utilidad para la labor docente y esperamos que su lectura sea inspiradora para la creación de nuevas prácticas.

José Pedro Derrégibus

Director de Fundación Telefónica – Movistar

Prólogo – Roberto Balaguer

La Comisión Europea en su informe de octubre de 2015 bajo el título "Computing our future. Computer programming and coding" European Schoolnet (2015) planteaba que:

"Las competencias y habilidades digitales son una de las principales condiciones para que la transformación digital de Europa sea un éxito, así como para su crecimiento y el bienestar de sus ciudadanos y sociedades (...) El reto para el sector educativo es elevar el nivel de dichas habilidades digitales en la futura fuerza de trabajo; pero, aún más importante, empoderar a la gente joven con competencias que les permitan dominar y crear sus propias tecnologías digitales, y prosperar en la sociedad actual. Creemos que la enseñanza-aprendizaje del 'coding', tanto en contextos formales como no formales, jugará un papel fundamental en este proceso."[1]

Aunque no tengamos aún una definición certera del término Pensamiento Computacional, haber comenzado a trabajar en nuestro Proyecto sobre la temática en Fundación Telefónica Movistar el mismo año que la Comisión Europea llevó a cabo ese planteo, tiene a mi juicio varios significados. Uno, ineludible, es la apertura y la visión de quienes dirigen este proyecto: José Pedro Derrégibus, María Noel Orellano, Beatriz Ríos y muchos más. Dos, el saber escuchar de parte de quienes toman las decisiones. Tres, entender que el mundo nos permite adelantar tiempos y brindarles a nuestros niños, niñas y adolescentes las mismas posibilidades que la vieja Europa se está planteando hoy para sus ciudadanos. Cuatro, confianza junto al trabajo con responsabilidad y esmero. Todo eso sucedió en este hermoso proyecto que hoy se corona con esta publicación. Tras dos años de trabajo, surge la presente publicación y aún tenemos preguntas: ¿cómo definimos pensamiento computacional?, ¿en qué consiste ese tipo de pensamiento?, ¿qué efectos cognitivos tiene en niños y adolescentes?, ¿cuál es su potencial? Aún hoy la comunidad internacional tiene un sinnúmero de preguntas sin una respuesta categórica. Diversas publicaciones abordan la cuestión del pensamiento computacional buscando definir a qué se refiere ese nuevo concepto en boga.

[1] European Schoolnet (2015). *Computing our future. Computer programming and coding: priorities, school curricula and initiatives across Europe* [Informe técnico]. Recuperado de http://www.eun.org/c/document_library/get_file?uuid=3596b121-941c-4296-a760-0f4e4795d6fa&groupId=43887

La programación, el coding o lo que también llamamos codigoalfabetización es más potente que el mero uso de herramientas. Factiblemente será una de esas habilidades necesarias para poder ser más proactivos en el mundo y, por tanto, menos dependientes. Entonces, no se trata solo de mejorar las habilidades de pensamiento sino de algo más crucial, que Rushkoff (2010) desde una posición sociocrítica que compartimos sintetiza en una frase:

"Programar para no ser programados."[2]

Las posibilidades de desarrollo del pensamiento lógico y de resolución de problemas, conjuntamente con las de mejora en la comunicación y el intercambio de experiencias, son elementos que pueden hacer la diferencia para una niña o niño que nace en un contexto de privaciones. El denominado pensamiento computacional o algorítmico supone un modo de abordaje y resolución de problemas, sean estos del tipo que sean, que empoderan a los chicos.

El pensamiento computacional ayuda a tomar decisiones de una manera ordenada, secuenciada, lógica, sin ambigüedades, algo que a veces resulta difícil de observar en el ámbito de las ciencias de corte más social. Aprender a programar empodera al sujeto frente a un mundo lleno de objetos digitales programados por otros.

Se trata de herramientas que les permitirán ser ciudadanos activos del futuro.
Si la alfabetización en lectoescritura fue el producto de las necesidades de la industrialización, las máquinas actuales necesitan de programación, de usuarios con nuevas alfabetizaciones capaces de llevarlas a su máxima potencia.

En el futuro, se verá la fusión de las inteligencias de las máquinas con las humanas. Habrá robots utilizando inteligencia artificial y tomando decisiones que afectarán a las personas.

Estamos entrando en una nueva era cognitiva donde los objetos (el denominado IoT o Internet de las cosas) y las personas pasan a estar, por defecto, conectadas. Las máquinas están aprendiendo, pero no todas las personas están aprendiendo a manejarlas. Saber manejarse eficientemente con los datos y los lenguajes de programación puede hacer la diferencia.

[2] Rushkoff, D. (2010). *Program or be programmed.* New York: OR Books.

Tenemos enormes desafíos estructurales por delante y el pensamiento computacional seguramente no será capaz de resolverlos a todos, pero al menos le dará a nuestros niños, niñas y adolescentes herramientas que los harán sentir menos indefensos, mejor provistos para enfrentar esos desafíos. El pensamiento computacional es una puerta de entrada a ese mundo del futuro ya presente.

Con proyectos como este les acercamos a nuestros niños, niñas y adolescentes la posibilidad de ser protagonistas de este mundo. Por eso, saludo con enorme alegría esta publicación y agradezco profundamente haber sido parte de este maravilloso trabajo.

Mag. Roberto Balaguer
junio, 2017

Prólogo – Gurises Unidos

En el transcurso del siglo pasado Uruguay muestra un alto grado educativo de su población como consecuencia de la temprana universalización de la educación primaria gratuita. Este rasgo distintivo contribuyó a la consolidación de una identidad nacional y a una idea de ciudadanía que fue sinónimo de escuela pública. En la actualidad existe un consenso amplio acerca de los límites del sistema educativo uruguayo con las particularidades que se manifiestan en cada uno de los subsistemas de la educación. Sin embargo, no hay un acuerdo sobre qué cosas deberían ser modificadas y sobre cómo deberían implementarse estos cambios.

Una educación con perspectiva de derechos supone garantizar que las prácticas no vulneren derechos básicos: la abolición de toda forma de violencia y el respeto a las identidades culturales. Esta perspectiva además implica entender a los niños y adolescentes como sujetos activos responsables y protagonistas, porque educar es educarse a uno mismo, en la interacción con otros y con el mundo.

Mientras, las tecnologías cambiaron la educación en dos de sus componentes esenciales: la socialización y los aprendizajes. El desafío planteado es cómo los sistemas educativos logran conectar con estos cambios y si -en caso de hacerlo- los utilizan a su favor para alcanzar los objetivos de una educación orientada al desarrollo integral de los niños, niñas y adolescentes.

Desde hace más de quince años Gurises Unidos y Fundación Telefónica - Movistar tienen una alianza estratégica que ha permitido realizar distintos aportes para transformar la realidad de niños, niñas y adolescentes en Uruguay. Esto nos ha impulsado a seguir diseñando dispositivos educativos innovadores como lo es el proyecto de Pensamiento Computacional. Incluir esta temática exigió un desafío extremadamente difícil que afrontamos junto con la Fundación y que hoy se resume en este trabajo.

Confiamos en que esta publicación sea un aporte para docentes y educadores interesados en el desarrollo de propuestas que incluyan a las tecnologías como un elemento de motivación para los aprendizajes, favoreciendo las acciones orientadas a transformar la educación de niños, niñas y adolescentes.

Lic. en Ps. Gonzalo Salles.
Director de Gurises Unidos.

ÍNDICE

INTRODUCCIÓN

Desde hace más de quince años Gurises Unidos y Fundación Telefónica - Movistar tienen una alianza estratégica que ha permitido realizar distintos aportes para transformar la realidad de niños, niñas y adolescentes (NNA). Eso nos ha impulsado a seguir pensando, diseñando y proponiendo enfoques educativos innovadores.

La propuesta de Pensamiento Computacional (PC) se encuentra orientada a la resolución de problemas desde el punto de vista ingenieril, lo que implica la incorporación de una serie de habilidades y competencias útiles para la búsqueda de soluciones y el desarrollo personal.

Es imprescindible mencionar y agradecer a los centros educativos y docentes que recibieron la propuesta, brindándonos el espacio, comprometiéndose y aportando en la implementación. En el año 2015 desarrollamos el proyecto en las escuelas N°299, N°119, N°63, y en el liceo N°42 de Montevideo; y la Escuela N°181 de Las Piedras, Canelones. En el año 2016 seguimos trabajando con la escuela N°299 y se sumaron a la propuesta el liceo N°65 y el N°33, la UTU Malvin Norte, el centro juvenil Santa María y el aula comunitaria N°5 de Montevideo.

En el capítulo 1 desarrollamos ideas sobre lo que entendemos son algunos desafíos en educación, en el capítulo 2 profundizamos en el concepto de educación desde una perspectiva de derechos. En el capítulo 3 definimos Pensamiento Computacional para luego identificar en el capítulo 4 cómo la propuesta podría contribuir a una educación de calidad. El capítulo 5 incluye los asuntos y actividades en programación, robótica y electrónica que proponemos para el trabajo con NNA. Por último, exponemos las lecciones aprendidas y conclusiones como aporte para el desarrollo de nuevas propuestas educativas en la temática.

1. Apuntes sobre algunos desafíos de la educación

Los desafíos de la educación tienen escala global. En Uruguay parece haber un acuerdo sobre estos y la necesidad de introducir acciones que los aborden. Sin embargo, no existen consensos amplios sobre la orientación que deberían tener estos cambios.

El presente documento pretende ser un aporte a la educación y para eso nos proponemos volver a pensar una pregunta clave: ¿cómo aprenden los niños, niñas y adolescentes?

Para contestar esta pregunta es necesario investigar y analizar las formas en que estos conocen el mundo y a sí mismos, es decir, las formas en que aprenden. Durante el desarrollo del proyecto hemos comprobado que tanto el interés como la motivación son claves para la adquisición de aprendizajes significativos.

Desde nuestro punto de vista la educación tiene que enfocarse en los NNA y en sus aprendizajes. Nuestra propuesta de incluir el trabajo de Pensamiento Computacional en el aula pretende ser un insumo que colabore en el diseño de nuevas prácticas educativas innovadoras.

1.1. Sobre las transformaciones de las nuevas tecnologías

Mientras tanto transitamos por una revolución de las Tecnologías de la Información y la Comunicación (TIC) que le dan al mundo en que vivimos un sello particular de época. A nivel global, la expansión de las tecnologías transforma distintas esferas de la sociedad, así como las actividades de los individuos.

Estas transformaciones van desde la comunicación, la producción de conocimiento, la circulación de la información, el vínculo de los gobiernos con los ciudadanos, el mundo del trabajo y la cultura, pasando por las relaciones cotidianas entre las personas.

En el año 2016, el 83% de los hogares uruguayos tenían acceso a internet. Son las generaciones más jóvenes las que hacen mayor uso de las nuevas tecnologías. El uso de internet decrece a medida que aumenta la edad. Mientras que el 96,7 % de los adolescentes y jóvenes entre 14 y 19 años son usuarios de internet, el 71% de los adultos entre 50 y 64 años lo son (AGESIC, 2016).

Algunos autores han planteado la distinción entre quienes nacen y se desarrollan en esta coyuntura denominándolos “nativos digitales”, diferenciándolos de quienes

transitaron la etapa de la niñez y la adolescencia sin convivir con estas herramientas tecnológicas, los llamados "inmigrantes digitales" (Prensky, 2010). Convivir o no en estos entornos tecnológicos, usar o no dispositivos digitales y redes sociales, parecen ser factores que justifican la distinción. Sin embargo, en los últimos años desde distintos ámbitos se cuestiona el término "nativos digitales". Este puede llevar a suponer que los más jóvenes tienen incorporadas habilidades y competencias innatas en el uso de las tecnologías. En la práctica, esta afirmación muchas veces se contrasta con la realidad.

La llegada de las tecnologías incide en una serie de cambios profundos sobre los pilares en los cuales se estructuró la educación: los modos de socialización y aprendizaje. Los NNA interactúan, se comunican y aprenden a través de diversas formas que incluyen de manera significativa el uso de las tecnologías.

Por otra parte, su uso masivo deja de manifiesto al menos dos cosas que ya sucedían pero que no eran visibles. Por un lado, la cantidad de formas de conocer y aprender que se dan por fuera de los centros educativos y por el otro, los desafíos de los centros para conectar con la realidad de sus alumnos.
Los sistemas educativos deben hacer un esfuerzo para incorporar estos cambios y utilizarlos para alcanzar los objetivos de una educación orientada al desarrollo integral de los NNA.

1.2. La brecha digital y los desafíos para la educación

El concepto de brecha digital se ha popularizado como la diferencia que se produce en el tener acceso o no a las tecnologías y a la infraestructura que garantiza, entre otras cosas, conectividad.

En el ámbito educativo, Uruguay desarrolló una de las políticas más innovadoras de los últimos años, el Plan Ceibal. Este se enmarca dentro de la iniciativa que a escala global desarrolló la organización *One Laptop per Child*[3] y buscó que los estudiantes y docentes de los centros educativos públicos de todo el país accedan a una computadora portátil, modelo "uno por uno".

En este contexto, encontramos a la educación pública y a Uruguay en un buen lugar de acceso a las nuevas tecnologías, reduciendo la brecha digital. Sin embargo, esto no garantiza que sea utilizado en el ámbito educativo de forma óptima y pertinente.

Las tecnologías como soluciones, utilizadas de manera adecuada, nos facilitan y potencian una serie de actividades vinculadas a los aprendizajes, abriendo nuevas oportunidades. Utilizarlas nos posiciona mejor frente al acceso y el manejo de la información, aunque para ello es necesario incorporar una serie de habilidades y capacidades que nos permitan conocer y usar adecuadamente las infotecnologías. Este proceso ha sido denominado "alfabetización digital".

Por otra parte, el potencial factor de cambio que tienen las tecnologías en la educación es la posibilidad que brindan a los estudiantes de pasar de usuarios a creadores, convirtiéndose en sujetos activos en la creación de nuevas soluciones. El proyecto *Robotic-Pensamiento Computacional* tiene como objetivo que los participantes se apropien de las tecnologías para la producción y el desarrollo de nuevos problemas y nuevas soluciones.

[3] http://one.laptop.org/

2. Educación con perspectiva de derechos

Las ideas y las creencias que predominaron en la sociedad consideraban a la niñez y la adolescencia como un sujeto pasivo de la educación. En la actualidad existen instituciones educativas amparadas bajo el marco de orientación disciplinaria y tutelar, donde el educando es objeto del derecho a educar. Por otro lado, surgen instituciones que empiezan a transitar un nuevo marco que incorpora la perspectiva del educando como sujeto activo del derecho a la educación, donde además se concibe a esta como acción pública orientada a la construcción de la ciudadanía [4].

Esto se encuentra vinculado al intenso trabajo de educadores y educadoras -entre otros- que han planteado la necesidad de ubicar en el centro de la educación a niños, niñas y adolescentes. Como observaba hace más de cincuenta años la maestra Reina Reyes, "...este cambio en cuanto al titular de derecho en el dominio de la educación, se proyecta en la concepción de los fines educativos. En efecto, si quien educa tiene derecho a hacerlo, establece los fines de la educación según su particular orientación política, filosófica o religiosa, en tanto que si el titular de derecho es el educando, los fines de la educación tienen que ser concebidos en función de los derechos de este, con el mayor respeto a sus condiciones naturales y proporcionando la libre elección de los valores complejos, en los dominios de la religión, la filosofía o la política."(Reyes 1964).

Por lo tanto, una educación con perspectiva de derechos supone garantizar que las prácticas educativas no vulneren derechos básicos (abolición del castigo físico, respeto a las identidades culturales, entre otras). Además impone una orientación de sentido para dichas prácticas: la formación del ciudadano, un sujeto con determinadas cualidades. Es partícipe, activo, responsable, crítico, informado; toma decisiones y acciones con otros, respeta las opiniones e identidades de los demás a la vez que promueve la participación y exige el respeto de los derechos humanos.

En este sentido, como veremos a lo largo de la publicación, trabajar desde nuestro proyecto con un enfoque de derechos implica:

[4] Los instrumentos de DDHH aplicados a la infancia consagran a la educación como un derecho y comprometen a los Estados Parte a garantizarla, no solo en términos de acceso sino también de calidad. Declaración Universal de Derechos Humanos, 1948. http://www.ohchr.org/EN/UDHR/Documents/UDHR_Translations/spn.pdf,

Convención Internacional sobre los Derechos del Niño, 1989. https://www.unicef.org/uruguay/spanish/CDN_20_boceto_final.pdf.

- posibilitar y favorecer la integración social:
    - · ser accesible
    - · favorecer la circulación social y el encuentro con otros
    - · ampliar el espectro de trayectorias posibles y generar oportunidades
- reconocer y promover las diversidades:
    - · de carácter, gustos, intereses
    - · de capacidades y potencialidades
- atender a la integralidad de la persona humana:
    - · posibilitar un abordaje comprensivo de la persona en un contexto comunitario protector y amigable
    - · propender al desarrollo de todos los aspectos de la persona
- gestionarse de manera participativa y autónoma:
    - · en los aspectos que refieren a la organización y la convivencia.
    - · en lo que respecta a los aprendizajes.

El proceso educativo debe ser una oportunidad para incorporar habilidades y competencias que nos permitan superar los obstáculos y problemas a los cuales nos enfrentamos.

Nuestra propuesta aborda una metodología centrada en el aprendizaje basado en proyectos, la cual consideramos una oportunidad para trabajar la dimensión individual y grupal de los aprendizajes. Además, tiene en cuenta el interés y la motivación de los NNA y conecta con su realidad. A partir de esta tríada promueve la formulación de problemas.

La educación tradicionalmente ha estado abocada a la presentación de soluciones, no de problemáticas. Los problemas y las soluciones ya estaban predefinidos. El desarrollo del PC no se centra en el uso de las tecnologías -soluciones- , sino que busca producirlas, es decir solucionar problemas. En este sentido la propuesta del PC pretende pasar de la transmisión de soluciones a la incitación de problemas. Un problema es un hecho real o imaginario que incomoda a nuestro intelecto produciendo la necesidad de una solución. Los problemas nos exigen una solución que nos mueve a crear y conocer, con otros que también tienen preguntas e intereses diversos.

Estos son movimientos claves para generar una ciudadanía activa, participativa, crítica, responsable, inclusiva y comprometida con su entorno.

3. ¿Qué entendemos por Pensamiento Computacional?

Los cambios y posibilidades que ofrecen las tecnologías de la información y comunicación generan un atractivo especial en las personas. En el ámbito educativo abren un campo de posibilidades nuevo y nunca visto. La posibilidad de manipular objetos, transformarlos y crearlos, convertir una idea en una acción, son oportunidades potentes para facilitar la adquisición de habilidades y la resolución de problemas.

La potencia de esta temática no está solo en el conocimiento acumulado detrás de las tecnologías, sino en la oportunidad para evidenciar los problemas que han llevado a estas soluciones y especialmente a otros para los cuales aún no se tiene solución.

3.1. Pensamiento Computacional y resolución de problemas

Con el objetivo de conceptualizar el PC realizamos una revisión bibliográfica y queremos resaltar las siguientes definiciones:

• El PC implica "modelos mentales que necesitamos para entender cómo resolver problemas a través de los computadores" (Bits, p.49 Cristian Bravo-Lillo).

• El PC es un "método para resolver problemas usando tecnología y está inspirado en el conjunto de competencias y habilidades que un profesional utiliza cuando crea una aplicación computacional" (Bits, p.30, citando a Wing 33-35).

• El PC es un tipo de "pensamiento abstracto-matemático/pensamiento pragmático ingenieril", es una "forma de resolver problemas de manera inteligente e imaginativa; combina abstracción y pragmatismo" (p.4, Valverde).

Resolución de problemas

En las diferentes definiciones propuestas se hace énfasis en la resolución de problemas como la esencia del PC. Ahora bien, ¿qué entendemos por la capacidad de resolución de problemas?

"La resolución de problemas se define como la capacidad de participar en un proceso cognitivo para entender y resolver problemas donde no hay un método de solución inmediatamente obvio" (OECD 2014 PISA in Focus 38); en otras palabras, un proceso activo de producción y aprendizaje.

Según el trabajo de investigación "Resolución de problemas de química y estructuras cognoscitivas" realizado en la Universidad de Keele U.K. (KEMPA, p.101, 1987)

se distinguen dos formas principales de concebir la resolución de problemas y de entender su función. La primera implica procesos **productivos** de resolución de problemas, en los cuales la persona produce por descubrimiento una combinación de reglas aprendidas previamente que puede aplicar para obtener una solución a una situación o problema novedoso. La segunda implica procesos **reproductivos** en los cuales el conocimiento es simplemente recordado o aplicado en situaciones o problemas no novedosos.

El PC aborda la resolución de problemas como un proceso productivo, a través de propuestas pedagógicas centradas en problemas auténticos ajustados a la realidad de los estudiantes. Esto les permite enfrentarse a diferentes situaciones sin repetir soluciones predeterminadas. En base al interrelacionamiento de conocimientos previos, los estudiantes elaboran soluciones novedosas y ajustadas a las características de la situación que enfrentan.

3.2. Habilidades que desarrolla el Pensamiento Computacional

Nos detendremos ahora en las habilidades que este enfoque promueve. Para eso, destacamos la definición que propone *Toolkit* o Caja de herramientas para líderes de Pensamiento Computacional creada por la Asociación de Docentes en Ciencias de la Computación (CSTA por su sigla en inglés)[5] y la Sociedad Internacional para la Tecnología en Educación (ISTE por su sigla en inglés) . Plantea una definición abierta, enumerando algunas de las características principales del PC, que conjuga tanto habilidades cognitivas como actitudinales.

[5] La Asociación de Docentes en Ciencias de la Computación (CSTA) es una organización creada en 2004 que promueve la enseñanza de la Ciencias de la Computación. Hoy en día cuenta con más de 25.000 docentes asociados que representan a más de 145 países. Su sede central se ubica en la ciudad de Albany, en Estados Unidos. (www.csteachers.org).

La Sociedad Internacional para la Tecnología en Educación (ISTE) es una asociación de docentes comprometidos en promover el uso eficaz de la tecnología en la enseñanza y la formación de los educadores. Su sede se ubica en Arlington, en Estados Unidos. (www.iste.org).

El *Toolkit* señala que el PC es un proceso de solución de problemas que incluye (pero no se limita a) las siguientes características:

- Formular problemas de manera que permitan usar computadores y otras herramientas para solucionarlos.
- Organizar datos de manera lógica y analizarlos.
- Representar datos mediante abstracciones, como modelos y simulaciones.
- Automatizar soluciones mediante pensamiento algorítmico (una serie de pasos ordenados).
- Identificar, analizar e implementar posibles soluciones con el objetivo de encontrar la combinación, más eficiente y efectiva, de pasos y recursos.
- Generalizar y transferir ese proceso de solución de problemas.

Estas habilidades se apoyan y acrecientan mediante una serie de disposiciones o actitudes que son dimensiones esenciales del PC. Estas disposiciones o actitudes incluyen:

- Confianza en el manejo de la complejidad.
- Persistencia al trabajar con problemas difíciles.
- Tolerancia a la ambigüedad.
- Habilidad para lidiar con problemas no estructurados.
- Habilidad para comunicarse y trabajar con otros para alcanzar una meta o solución común (CSTA e ISTE, 2011).

En esta línea y haciendo acuerdo con la definición propuesta, describiremos algunas habilidades que en base a nuestra experiencia entendemos que se despliegan al trabajar el PC.

Habilidades socio-emocionales.

Para conocer, investigar y crear es necesario poner en juego una serie de habilidades socioemocionales básicas. El gusto y el interés por la temática resultan los principales impulsores de la curiosidad por aprender, ya que implican una motivación hacia las propuestas y los problemas a enfrentar. Alcanzar una buena disposición al aprendizaje exige lograr que los estudiantes perciban emociones y sentimientos de satisfacción frente a las propuestas del aula, lo cual no excluye que debamos trabajar sobre cómo tramitar adecuadamente sentimientos de ansiedad o frustración.

Cuando los problemas resultan desafiantes generan la oportunidad de desarrollar y fortalecer habilidades socio-emocionales: el autoconcepto y la autoeficacia para enfrentar el desafío de resolver un problema, la perseverancia y la tolerancia a la frustración para volver a intentarlo teniendo claro los objetivos, la comunicación asertiva y la empatía para el trabajo en equipo.

Lenguaje y comunicación.

Para controlar un robot, crear un juego o interactuar con un microprocesador es necesario aplicar diferentes lenguajes de programación y protocolos de comunicación. Este proceso exige a priori ejercitar las habilidades lingüísticas básicas para la comunicación. La programación requiere adecuarse a un código establecido, que conlleva una sintaxis y una semántica específicas, básicas en cualquier lenguaje de programación. Se debe ordenar, jerarquizar y sintetizar para garantizar la eficacia del código, habilidades imprescindibles en la programación y transversales a cualquier tipo de comunicación.

El lenguaje de código implica el uso de estrategias básicas de relación interpersonal e intercambio social. Lo importante es el aprendizaje que se adquiere a través del trabajo grupal que genera una comunicación asertiva y el intercambio de experiencias y opiniones que se pueden replicar en otros contextos. Este proceso conlleva la adquisición de habilidades como la escucha activa, el respeto a las opiniones, la reflexión, empatía y tolerancia.

Descomposición y deconstrucción de un problema.

El proceso de resolución de un problema tiene como punto de partida la formulación de él mismo. Parte de su resolución está en la manera en que lo formulamos. Que los estudiantes definan ellos mismos el problema hace que les resulte motivante y desafiante. Esto no implica que sea obvio o trivial, todo lo contrario, la imaginación y la creatividad llevan a proponerse problemas complejos. Por otra parte, para intentar comprenderlo es necesario deconstruirlo, es decir, iniciar un proceso de descomposición que nos permita dividirlo en pequeños problemas hasta el punto que la resolución de ellos sea sencilla o de menor dificultad.

El PC promueve conocimientos en diversas áreas temáticas dado su fuerte hincapié en el uso de tecnologías. Algunos ejemplos de estos conocimientos son: electrónica (necesaria para armar circuitos), matemática y lógica (necesarias para elaborar algoritmos en la programación), armado de modelos (aplicado en el montaje de robots con piezas de un kit), creación de sistemas (armado de robots con objetos de nuestra cotidianidad). En el capítulo 5 “Asuntos y actividades” se desarrollan propuestas para trabajar estas unidades temáticas.

3.3. Problemas comunes

¿Qué similitudes encontramos entre ayudar a una persona perdida, hacer un mapa semántico y construir un robot?

A la hora de ayudar a una persona perdida para llegar a un lugar se necesita primero **recopilar** la información necesaria, como el punto de partida y de llegada, así como puntos de referencia compartidos con el interlocutor. Es útil también recrear la información analizada **representándola** en un esquema para poder luego optar por el camino más apropiado. Una vez visualizado el mejor camino, es necesario **descomponerlo** en tramos menores para facilitar la explicación. Después de recreado el recorrido mentalmente se deben organizar las instrucciones para poder **elaborar un algoritmo** del trayecto a través de una secuencia ordenada de pasos a seguir.

Es importante tener en cuenta que a la hora de comunicarle al otro el algoritmo a seguir, como mencionamos anteriormente, es necesario establecer una **comunicación asertiva** que nos permita entablar un diálogo fluido con el interlocutor. Se debe manejar un código preciso, claro, objetivo y compartido por el otro, que permita la comprensión cabal del recorrido y por ende la llegada al punto de destino de la persona perdida.

Al momento de elaborar un mapa semántico sobre un concepto, debemos comenzar **recopilando** la información relevante, para luego **analizar** los datos, encontrarles sentido y **jerarquizarlos.** Una vez extraídas las ideas principales, se debe **estructurar** la información en categorías, y relacionar estas con el concepto principal. Por último, se deben **representar** los datos gráficamente, escogiendo las palabras y la estructura más adecuada para poder sintetizar y ordenar las ideas fundamentales.

Para construir un robot que cumpla una función determinada, como seguir una línea o esquivar objetos, se debe **analizar el problema** para poder **abstraer** y reducir su complejidad. Es preciso **simular** las diferentes situaciones que este deberá enfrentar, **analizando y recopilando** datos que permitan deducir sus rasgos esenciales. Luego **descomponer** el problema, dividiéndolo en partes más pequeñas y manejables, en el que cada miembro del equipo pueda aportar desde sus potencialidades. Es necesario **proyectar** las diferentes partes como fracciones de un sistema, garantizando su ensamblaje, conformando una unidad que realice la función esperada. Se debe además **verificar** su real funcionamiento, una y otra vez, para ajustar posibles errores que serán corregidos a lo largo de reiteradas pruebas que permitirán mejorar el diseño y sus algoritmos de programación.

Al momento de enfrentarnos a un problema seguimos una serie de pasos que ordenan y orientan la resolución. Comenzamos analizando la situación a enfrentar, definimos y formulamos el problema, construimos posibles soluciones, evaluamos la más adecuada, culminamos verificando y poniendo en práctica la solución. La confianza y la perseverancia nos permiten enfrentarnos a cualquier situación, desarrollando nuestras competencias socio-emocionales. Entendemos que enfrentarnos a problemas de las ciencias de la computación (como diseñar y armar un robot) nos permite ejercitar habilidades vinculadas a la resolución de problemas. Estas habilidades nos posicionan mejor a la hora de enfrentarnos a otras situaciones de la vida cotidiana con las mismas herramientas.

4. El Pensamiento Computacional como oportunidad

4.1. Como un factor de alfabetización

Trabajar el PC nos posibilita en primer lugar responder a un contexto determinado en el cual el manejo de las nuevas tecnologías se vuelve tan importante como aprender a leer y escribir. Como sostiene una de las creadoras del concepto de PC, Jeannette Wing, "*To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability*"[6] (Wing, 2006). Trabajar con esta temática no solo posibilita conectar con la realidad que viven los estudiantes, sino que también promueve una relación activa y crítica con el entorno tecnológico. El PC trasciende el campo de la alfabetización digital y permite comprender qué es lo que ocurre detrás de todo dispositivo tecnológico. Esto favorece un mejor uso de la tecnología, así como también posibilita a los NNA ser creadores de tecnología y no solamente hábiles usuarios.

Trabajar PC no solo nos posibilita incidir en la calidad de la educación desde una perspectiva de derechos, sino que también posibilita trabajar con una metodología diferente, caracterizada por la elaboración de proyectos.

4.2. Potenciando el trabajo colectivo

Trabajar en base a proyectos permite potenciar el trabajo en equipo, generando un marco para la colaboración, donde cada persona participa en función de sus intereses, sus motivaciones y sus capacidades, apoyándose en el trabajo que realizan otros, en la búsqueda de un objetivo común. Los proyectos de ingeniería tienen la virtud de presentar muy diversos planos de trabajo que exigen la puesta en juego de variadas habilidades, generando oportunidades para que todos los participantes puedan verse reflejados en el producto.

[6] A las habilidades analíticas que todo niño desarrolla como la lectura, escritura y aritmética, deberíamos agregar la de Pensamiento Computacional.

Por otra parte, la esencia misma del PC que, como apuntábamos más arriba, se presenta en la formulación y abordaje de los problemas como primer paso en la búsqueda de posibles soluciones, constituye un campo fértil para el trabajo en equipo.

En primera instancia es necesario que miremos el problema de diferentes maneras, que pensemos lateralmente o *fuera de la caja.* Muchas veces no se encuentra una solución al problema no por su complejidad intrínseca o por la ausencia de medios o recursos para resolverlo, sino por la forma en que inicialmente se nos presenta. Es aquí donde los grupos que logran considerar muchos puntos de vista, sin censurar o cohibir a sus participantes, se encuentran fortalecidos y potenciados en comparación con un abordaje individual de los problemas. Aquí es importante que la figura del líder pedagógico (el educador) se encuentre atenta a desarticular algunas actitudes de censura o la creencia de que hay una sola forma correcta de resolver determinado problema. Debe propiciar la duda y la argumentación abierta a la existencia de otros puntos de vista.

Luego de construido el problema, trabajar en su resolución implica descomponerlo en pequeños problemas que puedan ser resueltos de manera más evidente. Con esto estamos generando un campo de trabajo colectivo, donde diferentes arreglos dentro del grupo pueden avanzar paralelamente, siempre atentos a las soluciones logradas por los demás en las otras partes o aspectos del problema general.

4.3. Formando parte de una comunidad científico-tecnológica global que comparte los problemas y los conocimientos: Sobre hombros de millones

¿Quién inventó la computadora? Esta pregunta planteada de manera ingenua nos abre la puerta a uno de los aspectos más característicos de las soluciones de ingeniería en general, y muy particularmente de los desarrollos en ciencias de la computación. El dispositivo tecnológico que usted puede tener sobre el escritorio o en un bolsillo es el resultado de la resolución de una inmensa cantidad de problemas. Estos no fueron abordados de una vez por alguien, sino por la acumulación a lo largo del tiempo de una enormidad de soluciones obtenidas por muchas personas en distintos lugares del mundo, que hicieron posible concebir, diseñar y construir masivamente cada componente de hardware y software.

Determinar un número aproximado de inventores detrás un artefacto de alta tecnología puede resultar una tarea titánica si no imposible, pero considerando la cantidad de componentes involucrados en el funcionamiento de una computadora actual es perfectamente factible que ese número oscile en el orden de los millones ¿Cómo es posible que los descubrimientos y las invenciones de millones de científicos e ingenieros se sinteticen en un dispositivo que podemos comprar por unos cuantos dólares?[7] La respuesta a esta pregunta está en la comunidad científico-tecnológica.

La expansión exponencial de los desarrollos tecnológicos en la segunda mitad del siglo XX se sustenta en el hecho de que estos han posibilitado la construcción de una comunidad global cada vez más amplia y abierta, que comparte los problemas y las soluciones que se van obteniendo. Basta observar el desarrollo que ha tenido la industria del software, que ha desbordado las limitantes impuestas por la competencia comercial, transitando inexorablemente hacia un modelo de código abierto. Un proceso condicionado estructuralmente, ya que al hacer accesible a los usuarios interesados los códigos que están detrás de una implementación de software, esta se hace mucho más adaptable a diferentes circunstancias y tiene un desarrollo notablemente más rápido. Esto redunda en ventajas estratégicas para las firmas comerciales que optan por este modelo. Un movimiento paralelo se insinúa en los últimos años en la industria del hardware.

Ahora bien, ¿en qué forma un grupo de NNA puede colaborar con una comunidad científico-tecnológica desde una práctica que se enmarca en un proceso educativo? Existen varios lineamientos de trabajo que hacen posible compartir globalmente problemas y soluciones.

Las competencias de Robótica, también llamadas deportes robóticos, son un claro ejemplo de comunidad global compartiendo problemas[8]. De ellas han surgido muchos de los avances en ingeniería robótica que han permitido encontrar soluciones a problemas reales, como por ejemplo, rescatar a los sobrevivientes luego de un terremoto sin poner en riesgo la vida de los rescatistas.

[7] En 2011 la Fundación Raspberry Pi en el Reino Unido dio a conocer una computadora de una sola placa, que pasó a ser conocida como la computadora de 5 libras, por su bajo costo. El objetivo era generar una computadora versátil que pueda ser accesible para todos los niños, incluso en el tercer mundo, y que estimule el conocer que hay detrás de las tecnologías y posibilite a las personas construir sus propios dispositivos. Ver http://respberrypi.org

[8] Se trata de competencias donde diversos grupos de estudiantes o profesionales alrededor del mundo abordan un problema imaginario o planteado mediante el juego. Ver: RoboCup, SUMO.uy, First Lego League, etc.

Paralelamente, los diferentes lenguajes de programación se desarrollan gracias a las comunidades que comparten las soluciones para generar nuevas implementaciones. Esto se produce a través de plataformas que permiten el manejo de múltiples versiones de librerías y aplicaciones de código abierto[9].

4.4. Conectando los saberes teóricos con la práctica

Como apuntábamos más arriba, la principal potencialidad educativa del PC como campo temático radica en la posibilidad de crear tecnología. Es por eso que estos asuntos, conjuntamente con el abordaje mediante proyectos, van a contrapelo de una práctica educativa caracterizada por la exposición de saberes teóricos, frente a los cuales los estudiantes no se han realizado ninguna pregunta. Sin desestimar la importancia de tener acceso a dichos saberes teóricos, ni de la enseñanza expositiva como método (muchas veces pertinente), debemos poner de relieve la necesidad de promover los procesos de aprendizaje mediante el tránsito por experiencias concretas. Estas experiencias deben enfrentar a los estudiantes a problemas de orden práctico, es decir, aquellos que se interponen entre nosotros y lo que queremos hacer.

Cierto es que existen muchos problemas que habitan únicamente el universo de la teoría y que es pedagógicamente deseable que desarrollemos la curiosidad y el gusto por abordarlos, pero no podemos pretender que su simple exposición magistral desate interés y curiosidad donde no los hay. Para ello es imprescindible que partamos de las vivencias, como las que encontramos en nuestra vida cotidiana, pero también aquellas a las que podemos llegar mediante la imaginación; y es por esto último que el juego resulta una herramienta tan poderosa para activar los procesos de aprendizaje y creación.

4.5. Apostando a la no directividad

Es importante tener en cuenta que a la hora de trabajar el PC con los estudiantes debemos potenciar lo que Gilles Ferry, en su libro Nuevas actitudes en la relación pedagógica, ha llamado la "no directividad". Se trata de encontrar nuevos modos de relación con el estudiante donde prime una actitud comprensiva del otro. Se intenta encontrar, en una búsqueda compartida. los intereses de los estudiantes de manera de aumentar sus posibilidades de autonomía y de responsabilidad.

[9] Un ejemplo notable es Git, una aplicación que permite a los programadores acceder, multiplicar y compartir rápidamente versiones de código fuente. Ver: github.com.

Para ello, es necesario que el educador no se imponga con modelos y estructuras ya dados, donde, por ejemplo, todos deben aprender lo mismo y al mismo tiempo. Siguiendo esta línea de la "no directividad", es el estudiante quien en base a sus intereses, capacidades y motivaciones decide el camino y los tiempos de su aprendizaje, y el docente, con una actitud comprensiva, interviene para facilitarles a los alumnos la experiencia directa de los problemas.

4.6. Promoviendo la participación genuina y la autonomía

Pero lo que creemos aún más importante del trabajo en base a proyectos es todo lo que permite en cuanto a la apropiación del proceso y del producto realizado. En nuestra educación actual, tanto los objetivos como las tareas y los procesos suelen ser impuestos por el docente, rara vez se generan instancias en las cuales el alumno pueda elegir o proponer materias o metodologías de trabajo. Así, las necesidades no surgen de los alumnos sino de los docentes. Por lo tanto, es entendible que el proceso y el producto pierdan sentido para quien nunca los ideó desde su origen.

Con una metodología pensada en el trabajo en base a proyectos, apostamos a que los estudiantes se apropien del proceso y del producto de su trabajo, promoviendo un manejo autónomo y participativo, que responda a los intereses y a las motivaciones de los participantes. De esta manera, educadores y educandos trabajan juntos e intercambian sus roles compartiendo y colaborando con lo que cada uno sabe, en pos de un objetivo común, generando un nuevo compromiso con el proceso de aprendizaje propio y de los demás.

5. Asuntos y Actividades

A continuación, se presenta una propuesta didáctica que sistematiza el trabajo realizado por el proyecto en diferentes espacios de educación formal y no formal durante los años 2015 y 2016. En los capítulos anteriores se profundizó sobre la metodología y la fundamentación de los contenidos; en el siguiente desarrollamos una propuesta orientada a la promoción de habilidades y competencias en ciencias de la computación, el pensamiento ingenieril y la resolución de problemas.

La propuesta educativa es una lista de asuntos y actividades que los educadores han de tener presente como insumos para la planificación didáctica que no está estructurada como una secuencia única. Esta sistematización contiene una serie de conocimientos básicos y otros avanzados, el desafío durante el proceso educativo está en cómo los participantes transitan la propuesta articulando y conectando estos conocimientos.

Este trabajo es un mapa de actividades tentativo que busca ser un marco de referencia para desarrollar esta temática. Las expectativas de cada participante deberán ser interpretadas por los educadores y constituirán el principal insumo para el diseño del itinerario individual y grupal. Las secuencias de implementación de las actividades será el resultado de los intereses, los conocimientos previos de los participantes, sus evaluaciones continuas, la disponibilidad de materiales de cada centro, así como la experiencia y el conocimiento de los docentes que facilitan la propuesta.

5.1. Programación

Un programa o algoritmo es una lista de instrucciones claras, ordenadas, objetivas y precisas que describen un procedimiento para la solución de un problema. La programación es una herramienta para la creación de programas informáticos que realiza procedimientos de manera eficaz y eficiente, escritos en un lenguaje de programación que la computadora interpreta y ejecuta.

Existen diversos lenguajes de programación que permiten a través de la lógica y el análisis sistémico de los problemas secuenciar órdenes para procesar datos y en base a ellos realizar ciertas funciones.

5.1.1. Programación Unplugged o Desconectada

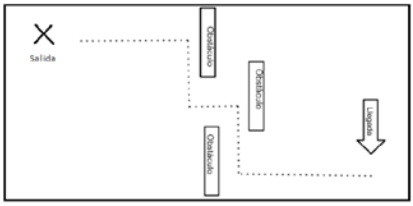
Para trabajar conceptos básicos de programación no es necesario utilizar una computadora. En ese sentido es que se habla de programación *unplugged*[10] (desconectada) sin la mediación de una computadora. La programación *unplugged* resulta una muy buena herramienta para introducirnos en el PC. Se trabajan conceptos fundamentales a través del juego, lo que permite abordarlos desde la experiencia. A su vez, resulta una oportunidad para establecer vínculos entre los participantes, posibilitando una relación educativa de confianza imprescindible en la elaboración de proyectos y en cualquier acto educativo.

[10] Nos basamos en la línea que desarrolla la Universidad de Canterbury de Nueva Zelanda, en colaboración con Google, quienes elaboraron un proyecto llamado "Computer Science Unplugged, Computer Science without a computer" en el cual comparten distintos recursos para trabajar programación. En su página web: http://csunplugged.org/ está disponible el libro "Computer Science Unplugged, Un programa de extensión para niños de escuela primaria" que contiene diversos recursos para enseñar programación sin computadoras de por medio. La última versión del libro es de marzo 2015 y está disponible en inglés, de lo contrario, se puede acceder también a la versión del libro traducida al español que data de diciembre de 2008:http://csunplugged.org/wp-content/uploads/2014/12/unpluggedTeachersDec2008-Spanish-master-ar-12182008.pdf

Nombre actividad	Jugando a los robots
Objetivo	Identificar la complejidad de elaborar una secuencia de instrucciones para hacer una tarea. Evidenciar la necesidad de ser precisos al crear una instrucción
Áreas de conocimiento	Matemática. Lengua. Ciencias de la computación
Conceptos relacionados	Secuencia, algoritmo, estructuras de control.
Competencias	Resolución de problemas. Programación.Trabajo en equipo Comunicación
Materiales	Hojas, lápices, tela para tapar los ojos.
Modalidad	Trabajo en subgrupos
Enlaces de interés	
Anexo	

Desarrollo de la actividad **Jugando a los robots**
El propósito de esta actividad es que los estudiantes identifiquen todo lo que tiene que ser tomado en cuenta a la hora de elaborar una lista de instrucciones para decirle a un compañero que se traslade desde un lugar a otro.

Identificar un punto de partida y un punto de llegada para que cada equipo elabore una secuencia de instrucciones que le permita a una persona con los ojos tapados trasladarse desde el punto de partida al punto de llegada asignado. Con el fin de complejizar la tarea, disponer en el espacio diferentes obstáculos que deberán ser sorteados por los participantes.



Para verificar la secuencia de instrucciones realizada por el equipo 1, dos voluntarios del equipo 2 leen y ejecutan la secuencia del equipo 1, reflexionando colectivamente de qué manera se pueden mejorar las secuencias realizadas haciendo hincapié en la importancia de elaborar instrucciones claras, ordenadas, precisas y objetivas. Es importante detenerse en los nudos que se presentan al experimentar la propuesta para que los integrantes de los equipos tengan la necesidad de buscar nuevas alternativas más eficaces para la ejecución de la secuencia.

Por ejemplo, es común que los estudiantes hagan referencia a la cantidad de pasos para señalarle a una persona cuánto debe caminar desde un punto hacia otro. A la hora de experimentar esa instrucción, verán cómo los pasos varían de persona a persona y, por lo tanto, tendrán la necesidad de reinventar esa orden, buscando una alternativa, como, por ejemplo: caminar hasta que... escuche a un compañero haciendo palmas. De esta manera, los estudiantes se van aproximando al concepto estructura de control.

El voluntario que ejecuta la secuencia solo realiza las órdenes que se le asignan sin poder preguntar nada, simulando ser una máquina. Se debe limitar a hacer lo que interpreta de las órdenes que su compañero lee. Los demás intervienen solo para cuidar a quien tiene los ojos cerrados. Repetir el procedimiento con todos los equipos.

Opcional: Proponerle a los equipos que rediseñen sus secuencias incorporando los insumos que surgieron en la reflexión, para luego verificarlas y visualizar los cambios.

Nombre actividad	Mis amigos robots. Actividad adaptada de “My Robotic Friends” desarrollada por Thinkersmith .
Objetivo	Introducirse en la programación y la codificación de instrucciones adaptándose a un lenguaje. Practicar la depuración de errores. Identificar la necesidad de utilizar funciones o procedimientos.
Áreas de conocimiento	Matemática. Ciencias de la computación.
Conceptos relacionados	Lenguaje de programación, secuencia, algoritmo, función, parámetro, depuración de errores.
Competencias	Resolución de problemas. Programación. Trabajo en equipo
Materiales	Hojas, lápices, goma, vasos de plásticos según participantes y el Anexo de Desafíos.
Modalidad	Trabajo en subgrupos
Enlaces de interés	http://programamos.es/web/wp-content/uploads/2014/10/MisAmigosRobot.pdf
Anexo	**Anexo 1 Desafíos**

Desarrollo de la actividad **Mis amigos robots**

El propósito de esta actividad es que los estudiantes elaboren una secuencia de instrucciones adaptándose a un código establecido, simulando el trabajo que realiza un programador. Y que además ejecuten una secuencia de instrucciones como lo realizaría un robot, siguiendo las órdenes al pie de la letra.

Se le propone a los estudiantes que elaboren un algoritmo para que otros compañeros puedan construir una estructura con vasos de acuerdo a un modelo prediseñado.

Los programadores solo podrán utilizar estos 5 símbolos para jugar:

↑ ↓ → ← ↻

Cada símbolo representa: subir vaso, bajar vaso, mover ½ ancho de vaso hacia la derecha, mover ½ ancho de vaso hacia la izquierda y dar vuelta el vaso respectivamente.

Para construir esta estructura de vasos, una posible solución es la siguiente:

Inicio, todos los vasos apilados.

1) ↑ → →↓ ← ←, colocamos el primer vaso.

2) ↑ → → → →↓ ← ← ← ←, colocamos el segundo vaso.

3) ↑ → → →↓, colocamos el último vaso.

La traducción de estos pasos sería:
Subir vaso, mover 2 pasos hacia la derecha, bajar vaso, mover 2 pasos hacia la izquierda para volver a la pila.
Subir vaso, mover 4 pasos hacia la derecha, bajar vaso, mover 4 pasos hacia la izquierda para volver a la pila.
Subir vaso, mover 3 pasos hacia la derecha, bajar vaso.

Quien ejecuta las instrucciones y asume el rol de "robot" debe colocar una pila de vasos sobre la mesa y realizar este procedimiento a un costado de la pila. Después del punto 1) al subir un vaso pasará automáticamente por encima de las filas de vasos ya construidas.

Cada equipo tendrá un voluntario que jugará el papel de "robot" y deberá salir del salón por unos minutos llevándose los vasos de su equipo. Los demás participantes o "programadores" de los equipos recibirán una imagen prediseñada de una estructura hecha con vasos (ver Anexo Desafíos). Deberán elaborar un algoritmo para construir esa estructura expresándola únicamente con el lenguaje establecido. Mientras tanto, se guiará al voluntario que está fuera en la ejecución de una secuencia de instrucciones a modo de prueba. Cuando el equipo termina de escribir las líneas de código, se llama a los robots que estaban fuera para que ejecuten la secuencia construida. Para ello, los robots se deben limitar a ejecutar exclusivamente las líneas de código escritas, sin hablar con el resto del equipo. Los programadores observarán los pasos incorrectos y si hay un error pueden parar la ejecución.
El robot tendrá que volver a salir y los programadores tendrán que depurar el algoritmo, para que luego el robot vuelva a ejecutarlo. Repetir el procedimiento avanzando en los desafíos y rotando al voluntario que juega el rol de "robot".

Los equipos que lleguen a los últimos desafíos deben buscar maneras de simplificar la notación, ya que al enfrentarse a desafíos más complejos la cantidad de flechas que debemos utilizar aumenta. Por esto es necesario utilizar notaciones que simplifiquen la elaboración de las instrucciones. Las flechas con un número entre paréntesis son una forma inteligente de indicar que queremos repetir la instrucción un número específico de veces. Se crea un nuevo símbolo para evitar repetir código innecesariamente. Esa es exactamente la idea detrás de las iteraciones, es decir, repetir una acción x cantidad de veces. Cuando asociamos un número llamado parámetro a la instrucción "mover ½ ancho de vaso hacia la derecha" → (3) estamos indicando que se repite tres veces esa instrucción.

A modo de ejemplo, se presenta una posible solución a un desafío para entender la utilización de las funciones.

1°- ↑→(2)↓ ←(2)
2°- ↑→(4)↓←(4)
3°- ↑→(6)↓←(6)
4°- ↑→(8)↓←(8)
5°- ↑→(10)↓←(10)
6°- ↑→(12)↓ ←(12)
7°- ↑↻→(3)↓ ←(3)
8°- ↑→(5)↓ ←(5)
9°- ↑↻→(7)↓ ←(7)
10°- ↑→(9)↓ ←(9)
11°- ↑↻→(11)↓.

Nombre actividad	Máquina de dibujar. Actividad adaptada de "Siguiendo instrucciones" desarrollada por Computer Science Unplugged.
Objetivo	Identificar las dificultades de dar y recibir instrucciones. Comprender la necesidad de especificar las instrucciones con exactitud, evitando las ambigüedades. Deducir los elementos que debe contener una instrucción para ser lo más exacta posible.
Áreas de conocimiento	Matemática. Lengua. Ciencias de la computación
Conceptos relacionados	Instrucción, secuencia, algoritmo
Competencias Comunicación.	Resolución de problemas. Programación. Diseño y creatividad.
Materiales	Lápiz, papel, regla y Anexo Dibujos para dictar.
Enlaces de interés	http://csunplugged.org/wp-content/uploads/2014/12/unpluggedTeachersDec2008-Spanish-master-ar-12182008.pdf (pág. 102)
Anexo	

Desarrollo de la actividad **Máquina de dibujar**

Se comienza la actividad discutiendo el contenido implícito de una orden común como pedirle a alguien que salga del salón. Se entiende que la persona tiene que caminar a la puerta, verificar si está abierta o cerrada, si está cerrada abrirla, para luego salir. Estos son algunos de los contenidos implícitos en esta orden. Los participantes pueden hacer una lluvia de ideas de todo lo que una orden representa.

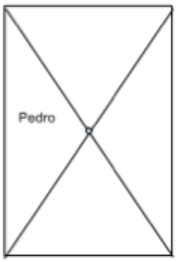
Luego se realiza un ejemplo de descripción de una imagen para explicar cómo se llevará adelante la actividad. Para eso los participantes tomarán una hoja y deberán interpretar y dibujar las siguientes instrucciones, sin dar posibilidad a preguntas. Se limitarán a interpretar lo que escuchan.

1. Pinta un punto en el centro de tu hoja.

2. Empezando en la esquina superior izquierda de la hoja, usando la regla dibuja una recta que pase por el punto y termine en la esquina inferior derecha.

3. Empezando en la esquina inferior izquierda de la hoja, usando la regla dibuja una recta que pase por el punto y termine en la esquina superior derecha.

4. Escribe tu nombre en el triángulo que está en el lado izquierdo de la hoja. El resultado debería verse más o menos así:



Luego de explicada la actividad se seleccionan dos voluntarios y se les proporciona una de las imágenes del enlace de interés sin que el resto del grupo la vea. Tendrán que describir la imagen para que los demás dibujen lo que interpretan de las instrucciones, sin realizar aclaraciones. Luego, se compara el original con los dibujos, evaluando grupalmente cuáles fueron las instrucciones que generaron ambigüedades y cómo se podrían mejorar. Al terminar pasan otros dos voluntarios a describir otra imagen, avanzando en la complejidad de las mismas y repitiendo el proceso, incorporando en su descripción las mejoras discutidas grupalmente.

Esta actividad se asemeja al proceso que se realiza al programar. La computadora ejecuta las instrucciones tal cual le fueron asignadas, sin aclaraciones. Solo se percibe el efecto de las instrucciones al culminar la ejecución de toda la secuencia. Este ejercicio resulta un entrenamiento para identificar la necesidad de ser precisos, claros y ordenados al momento de elaborar instrucciones o algoritmos.

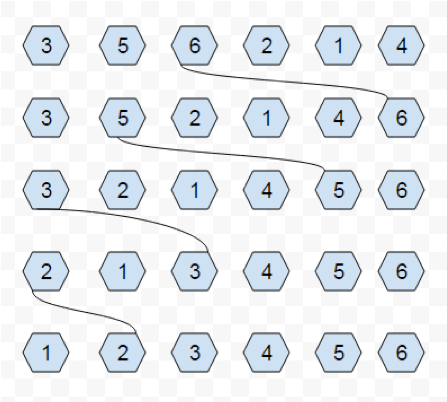
Nombre actividad	Algoritmos de ordenamiento, "Bubble Sort" Identificar diferentes posibilidades de ejecución de un algoritmo.
Objetivo	Practicar la depuración de errores. Estrategias como herramientas para la eficiencia. Utilizar el condicional SI.
Áreas de conocimiento	Lengua. Ciencias de la computación
Conceptos relacionados	Secuencia, algoritmo, estructuras de control, condición, depuración de errores.
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Comunicación
Materiales	
Enlaces de interés	
Anexo	

Desarrollo de la actividad **Algoritmos de ordenamiento, Bubble Sort.**

Se les plantea a los participantes que se suban a un banco y se ordenen alfabéticamente por el nombre, sin poder bajarse. Deberán idear maneras para comunicarse, organizarse y realizar el ordenamiento. Se puede cronometrar cuánto demoran en cumplir la consigna, para luego comparar los tiempos entre los diferentes métodos de ordenamiento. Se espera que aplicando una estrategia previamente discutida los tiempos disminuyan. Después de esta etapa se colectivizan las dificultades encontradas, así como los problemas de comunicación y organización. Se discute sobre cuál fue el método utilizado, para luego poder elaborar una estrategia que permita solucionar el desafío de manera más eficiente. Luego de probar y cronometrar los diferentes métodos ideados por los participantes, se puede plantear el siguiente algoritmo: Si quien está a mi derecha es menor que yo, me intercambio con él, si no, no.

Esta actividad está inspirada en un algoritmo de ordenamiento llamado Bubble Sort creado y utilizado en las ciencias de la computación para ordenar datos. Su adaptación consiste en que cada participante se compare una y otra vez con quienes están a su lado, intercambiándose con aquellos que corresponda. La lista estará completamente ordenada cuando no haya ningún intercambio posible al recorrerla.

A modo de ejemplo se ejecuta el algoritmo en una lista desordenada de 6 números. Se recorre la lista elemento por elemento comparando con el siguiente a la derecha. En la primera línea el proceso es el siguiente:



- se compara el 3 y el 5. El 5 es más grande entonces no se cambian.
- se comparan el 5 y el 6. El 6 es más grande entonces no se cambian.
- se comparan el 6 y el 2. El 6 es más grande entonces se cambian.
- se comparan el 6 y el 1. El 6 es más grande entonces se cambian.
- se comparan el 6 y el 4. El 6 es más grande entonces se cambian.

El método se repite en las sucesivas líneas hasta que no existan cambios posibles.

Se recomienda explicar el método con 6 alumnos, para luego ampliarlo a todo el grupo.
Se puede volver a cronometrar cuánto demoran en ordenarse con este método y comparar con los tiempos anteriormente cronometrados.

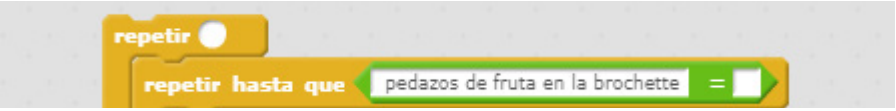
Luego en subgrupos se les puede pedir a los estudiantes que escriban el algoritmo que explica el método ejecutado. Por último, se comparten los diferentes algoritmos realizados buscando complementarlos para terminar creando uno solo, lo más exacto posible.

Nombre actividad	Cocinero robot
Objetivo	Identificar diferentes posibilidades de ejecución de un algoritmo. Practicar la depuración de errores. Ejecutar un algoritmo. Utilizar estructuras de control.
Áreas de conocimiento	Matemática Ciencias de la computación
Conceptos relacionados	Secuencia, algoritmo, estructuras de control, depuración de errores.
Competencias	Programación. Trabajo en equipo
Materiales	Frutas cortadas en trozos, palitos de brochettes, Anexo Receta.
Enlaces de interés	
Anexo	

Desarrollo de la actividad **Cocinero robot**

A cada equipo se le entrega una receta para elaborar brochettes de frutas (ver Anexo receta). Una receta es un algoritmo, es decir un conjunto ordenado de instrucciones que permite completar una tarea, en este caso preparar una bochette de frutas.

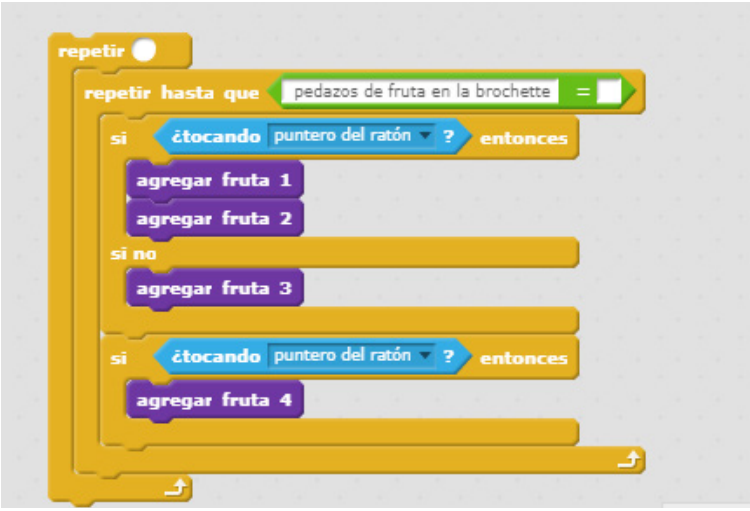
En esta receta se incluyen dos variables, las cuales son espacios que almacenan datos y que por tanto asumen un valor. En nuestro ejemplo "repetir" y "pedazos de fruta en la brochette" serían las dos variables a completar por los equipos.



Estas variables son los parámetros de repetición de las estructuras de control. El primer "repetir" está asociado a la cantidad de brochettes que realizará cada equipo y el segundo "repetir" está asociado a la cantidad de pedazos de fruta que tendrá cada brochette.

Este algoritmo también incluye los condicionales SI y SI_SI NO.
-SI pongo fruta 1 entonces agrego fruta 2, SI NO agrego fruta 3.
-SI tengo 2 o 4 pedazos de fruta colocados, entonces agrego fruta 4, SI NO vuelvo a repetir todo el procedimiento desde el inicio y me vuelvo a preguntar si la cantidad de pedazos de fruta en la brochette es igual a 5 (Si pedazos de frutas = 5, fin del "repetir hasta que", si no continúo la secuencia).

Las estructuras de control y los condicionales permiten modificar el sentido de los flujos de ejecución en función de ciertas condiciones. Este ejemplo de receta de brochettes dará unas combinaciones de frutas posibles y no otras. Estos conceptos también son trabajados en el recurso *Blockly Games* que se presenta en la siguiente unidad.



A cada equipo se le debe aclarar qué fruta corresponde a cada uno de los ítems: fruta 1, fruta 2, fruta 3 y fruta 4, según las variedades de frutas que se hayan conseguido para la actividad.

Por ejemplo:
fruta 1: banana
fruta 2: manzana
fruta 3: naranja
fruta 4: durazno

Los equipos tendrán que analizar el algoritmo, reconstruyendo la secuencia planteada para poder elaborar las brochettes que la receta propone.

Antes de la elaboración cada equipo deberá escribir en una hoja dos tipos de brochette que se pueden elaborar siguiendo el algoritmo de la receta y dos que no pueden ser elaborados si se respeta el algoritmo propuesto.

Un ejemplo de armado de una brochette con 5 pedazos de fruta es el siguiente:
Fruta 1, fruta 2, fruta 4, fruta 3, fruta 4.
Un ejemplo de una brochette que no se puede armar es el siguiente:
Fruta 1, fruta 3, fruta 2, fruta 3, fruta 4.

De esta manera, se pone en práctica la capacidad de simular la ejecución de un programa por parte de la máquina, para poder detectar posibles errores. Después de verificar en colectivo las distintas posibilidades de brochettes presentadas por el equipo, los integrantes las ejecutarán.

5.1.2. Juegos conectados para aprender a programar

Así como existen muchos recursos *unplugged* para aprender a programar, también hay disponibles en la web una gran variedad de recursos *plugged* (enchufados) donde los estudiantes pueden, mediante juegos de computadora, iniciarse en el mundo de la programación.

Los recursos *Lightbot* y *Blockly Games* que se desarrollan a continuación conjugan tres características que potencian su utilización. Estos son: el trabajo de conceptos y habilidades fundamentales para programar; el motivar a quienes los utilizan, transformando ejercicios lógicos en juegos llamativos cercanos a sus intereses y el impulso de la no directividad del aprendizaje, brindando la posibilidad de manejar sus tiempos y formas acordes a las capacidades, dando lugar a que el educador solo intervenga cuando exista una dificultad para resolver el desafío.

Como vimos, los algoritmos son un conjunto finito de instrucciones claras, ordenadas, objetivas y precisas para resolver un problema. Un algoritmo puede ser representado de varias maneras, con un texto, un diagrama de flujo, un lenguaje o un pseudolenguaje de programación. Las estructuras de control (Repetir, Hasta que, Mientras que) y los condicionales (SI, SI_SI NO) son instrucciones que permiten controlar el flujo de información dentro de un algoritmo determinado. Los recursos que presentamos a continuación tienen en común que trabajan algunos de estos conceptos y que además utilizan pseudocódigo o pseudolenguaje de programación.

El pseudocódigo es una alternativa para acercarse al lenguaje de programación combinando el lenguaje común con algunas convenciones o formalidades y está destinado a la comprensión humana. Es considerado de alto nivel porque se aleja del lenguaje máquina, lo que lo hace más accesible para comenzar a conocer ciertos contenidos de programación.

Nombre actividad	Lightbot
Objetivo	Practicar la elaboración de secuencias de instrucciones adaptándose a un pseudocódigo. Identificar la necesidad de utilizar funciones o procedimientos y ejercitar su uso. Introducirse a la noción de loop, bucle o ciclo.
Áreas de conocimiento	Ciencias de la computación
Conceptos relacionados	Problema, algoritmo, función, solución más eficiente, depuración de errores.
Competencias	Resolución de problemas. Programación
Materiales	Terminal y conectividad.
Enlaces de interés	https://lightbot.com/flash.html
Anexo	

Desarrollo del recurso **Lightbot**

El propósito del uso de este recurso es introducirse en el mundo de la programación, ejercitando la práctica de los conocimientos básicos trabajados en la unidad unplugged.
El objetivo del juego "Lightbot" es que el "Robot Luz" ilumine todos los cuadros azules que figuran en la pantalla. Quien juega debe indicarle al robot qué camino hacer para poder cumplir su objetivo. Para ello debe utilizar un pseudocódigo que contiene los siguientes comandos:

Cada comando representa: caminar hacia adelante, iluminar el cuadro, girar a la izquierda, girar a la derecha y saltar respectivamente.

En parejas o individualmente los estudiantes se enfrentan al recurso y van pasando las diferentes pantallas que el juego propone. Las pantallas están organizadas en tres niveles: Básicos, Procedimientos y Bucles.

Se espera que los estudiantes vayan haciendo su propio recorrido en el programa, brindándoles ayuda y orientación solamente en los momentos en que ellos lo necesiten.
El nivel 2 (Procedimientos) incorpora nuevos comandos a utilizar:

En las pantallas del nivel 2, los desafíos son más complejos y el uso de comandos se ve restringido, lo que exige la elaboración de nuevas estrategias. Se vuelve necesario usar nuevos comandos (P1 y P2) que permiten crear procedimientos, es decir, una lista de comandos que se repite, y puede ser llamada una y otra vez de la misma manera.

En el nivel 3 "Bucles" no se incorporan comandos nuevos. Sin embargo, la complejidad está en la restricción de la cantidad de comandos a utilizar. En este último nivel solo se permite la utilización de un comando, por lo tanto, se vuelve indispensable utilizar la técnica de programación llamada bucle. Esta misma consiste en la elaboración de un procedimiento que se repite indefinidamente, que culmina la secuencia invocándose a sí mismo

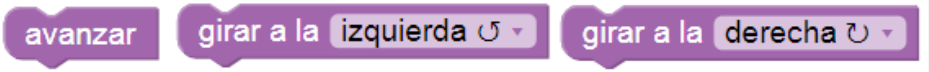
Nombre actividad	Blockly Games
Objetivo	Practicar la elaboración de secuencias de instrucciones adaptándose a un pseudocódigo. Profundizar la comprensión de las estructuras de control.
Áreas de conocimiento	Ciencias de la computación.
Conceptos relacionados	Problema, algoritmo, solución eficiente, depuración de errores, estructuras de control.
Competencias	Resolución de problemas. Programación.
Materiales	Terminal y conectividad.
Enlaces de interés	https://blockly-games.appspot.com/
Anexo	

Desarrollo del recurso **Blockly Games**

Blockly Games es un recurso educativo compuesto de distintos juegos que introducen diferentes técnicas y herramientas para la programación. Desarrollaremos el juego *Laberinto* en el que se trabaja con la función "Repetir hasta que" y el condicional "SI y SI SINO", aplicados a la construcción de algoritmos que solucionan los diferentes niveles. Para comenzar es recomendable utilizar el juego *Rompecabezas* (ver código QR) que introduce al pseudocódigo utilizado, llamado programación de bloques.

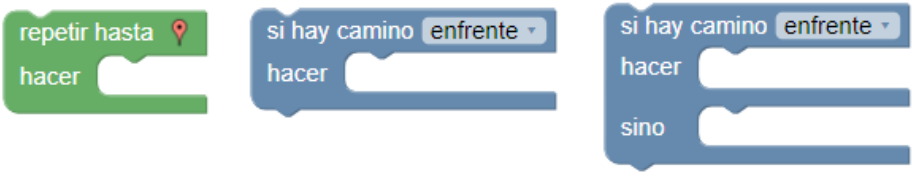
El objetivo del juego Laberinto es que el personaje logre alcanzar el punto marcado en el mapa. Quien juega debe construir un algoritmo que guíe al personaje en el movimiento por el mapa para poder llegar a su objetivo.

Para los primeros niveles los bloques a utilizar son los siguientes:



Existe una relación en torno al desafío planteado en este juego y la actividad Jugando a los robots. Las instrucciones que se utilizan en los primeros niveles son similares a las que intuitivamente se aplican en la actividad *unplugged.*

Al avanzar en los niveles del Laberinto se incorporan progresivamente estas estructuras:



El propósito del juego es entender y utilizar estos tres diferentes bloques, estructuras de control similares a las que permiten resolver los problemas de ambigüedad y precisión presentados en los algoritmos elaborados en la actividad Jugando a los robots.

Apuntes sobre juegos conectados

Es interesante la posibilidad de intercalar actividades de la unidad *unplugged* con la utilización de recursos *plugged* de manera de poder acercarse a los contenidos con múltiples estrategias. A continuación, se esbozan los conceptos fundamentales que se trabajan tanto en la unidad *unplugged* como en la *plugged*. La elaboración de secuencias de instrucciones es trabajada en las actividades Máquina de dibujar, Jugando a los robots y Mis amigos robots, así como en las primeras pantallas de *Lightbot* y *Blockly Games*. Las estructuras de control se introducen en las actividades Jugando a los robots, Cocinero Robot y son fundamentales para avanzar en los niveles de Blockly Games. El primer acercamiento a un pseudolenguaje se realiza en Mis amigos robots y Cocinero Robot. Las actividades *plugged* están expresadas a través de un pseudocódigo cuyo manejo es indispensable. Las funciones se utilizan en Máquina de dibujar y Cocinero Robot, así como en el Nivel 2 de *Lightbot*.

Si bien a modo de ejemplo se presentaron dos recursos didácticos web como *Lightbot* y *Blockly Games*, existe otra gran variedad de recursos *plugged* para introducirse en el mundo de la programación.

La organización *code.org* es una referencia a tener en cuenta ya que en su sección "la hora del código" hay disponibles más de 50 juegos para aprender a programar, pensados para diferentes niveles de acercamiento a la programación y diseñados atractivamente de acuerdo a los intereses de los niños y adolescentes.

El interés de trabajar con estos distintos recursos es poder aplicar los conocimientos adquiridos en los talleres unplugged. Se espera que los estudiantes vayan pasando los distintos niveles que los juegos proponen, haciendo su propio recorrido en el programa. Importa que ellos mismos identifiquen distintas estrategias para resolver los desafíos planteados, buscando reconocer la forma que mejor se adapta a cada uno. Es importante que los estudiantes puedan transitar solos por los recursos, de manera que cada uno vaya gestionando sus tiempos de acuerdo a sus necesidades, y que el educador oriente solamente cuando ellos lo necesiten. Es recomendable que a modo de cierre de cada actividad haya un espacio de socialización para que todos los participantes puedan compartir las distintas soluciones encontradas para resolver los desafíos. De esta manera se puede percibir la pluralidad de soluciones posibles para un mismo problema y a su vez se puede reflexionar sobre la importancia de generar estrategias que no solamente sean efectivas, sino que también sean lo más eficientes posibles. Las formas de entender y resolver los desafíos que construyen los participantes deben ser aprovechadas para resaltar la idea de que existen diversas estrategias válidas para resolver un problema.

5.1.3. Programación con Scratch

Scratch[12] es un lenguaje de programación desarrollado por el Dr. Mitchel Resnick del MediaLab del Instituto Tecnológico de Massachusetts (MIT por su sigla en Inglés) que permite programar de manera sencilla y atractiva juegos y animaciones. Es un programa gratuito y de software libre que posibilita introducirnos en el mundo de la programación de forma intuitiva, a través de una interfaz amigable. Se trata de una herramienta a tener en cuenta por los docentes de nuestro sistema educativo ya que es una aplicación disponible en todos los equipos del Plan Ceibal. A su vez, hoy en día es de fácil acceso ya que desde el 2013 el Scratch 2.0 es una aplicación disponible on line, en la que se pueden crear juegos y animaciones, compartiendo y colaborando con personas de todo el mundo.

Con Scratch es posible conectar los contenidos que se trabajaron en actividades previas: algoritmos, estructuras de control y funciones. Se aplican estos conocimientos para crear y editar juegos, a través del lenguaje de programación visual o de bloques. Scratch fue creado para ser amigable e intuitivo para el usuario. Este recurso permite, a diferencia de los anteriores, crear libremente diferentes proyectos multimedia interactivos, aplicando y ejercitando habilidades imprescindibles para programar.

Scratch brinda muchas más posibilidades que los recursos anteriores. Eso trae aparejado una mayor complejidad para la utilización de la interfaz. El asunto es no pensar que debemos "saber mucho" para utilizarla, ni conocer todas las herramientas, ni poder contestar todas las preguntas. El rol del educador en nuestra propuesta apunta a ser un andamiaje para utilizar la herramienta planteando preguntas y ayudando a descomponer y a resolver los problemas junto a los participantes.

[12] Esta unidad es producto de la exploración de la interfaz, la experiencia educativa en la temática, la participación en la formación online de la plataforma Coursera (https://es.coursera.org/learn/a-programar) llamada "¡A Programar!

Una introducción a la programación" y el manual de scratch "INFORMÁTICA CREATIVA" creada por Harvard Graduate School of Education (https://drive.google.com/file/d/0B1A2a6uTW0rfUlhFWG5uY19TWjg/view?usp=sharing)

Muchas de las computadoras del Plan Ceibal incluyen una versión de escritorio de Scratch. A pesar de esto es recomendable utilizar la versión online ya que respalda todas las creaciones, permite acceder desde cualquier lugar, compartir y ver los proyectos de otros usuarios. Para eso los participantes deben crearse una cuenta en el sitio de Scratch.

Creación de un usuario Scratch

Para la creación de un usuario Scratch cada participante debe en primer lugar acceder al sitio https://scratch.mit.edu/. Además, debe contar con un email; de no tener, puede utilizar el del educador o el de un compañero que sí tenga, ya que una dirección de email puede tener más de una cuenta asociada.

En la página de inicio del sitio, deben hacer click en “unirse a Scratch”.

Se despliega la ventana “Únete a Scratch”, en la que los participantes deberán crear un nombre de usuario y contraseña. Luego deberán completar algunos datos personales. Y, por último, realizar la verificación por email.

Nombre actividad	Visitando la biblioteca virtual de Scratch
Objetivo	Explorar las posibilidades de la herramienta investigando proyectos realizados por otros. Fomentar la búsqueda autónoma de información.
Áreas de conocimiento	Ciencias de la computación
Conceptos relacionados	Comunidad global, trabajo colaborativo.
Competencias	Programación. Investigación y experimentación.
Materiales	Terminal , conectividad y proyector.
Enlaces de interés	https://scratch.mit.edu/explore/projects/all
Anexo	

Desarrollo de la actividad Visitando la biblioteca virtual de Scratch

Desarrollo de la actividad **Visitando la biblioteca virtual de Scratch**

En la página oficial de Scratch existe una sección Explorar, donde miles de scratchers de todo el mundo comparten una infinidad de proyectos organizados en distintas categorías: animación, arte, juegos, música e historias.

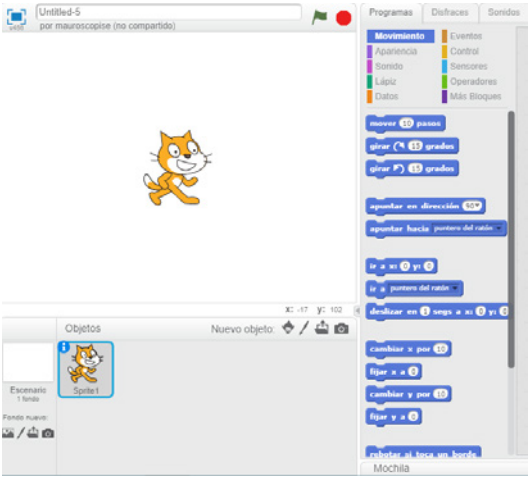
El interés de este primer acercamiento a la biblioteca virtual de Scratch es que los estudiantes puedan visualizar todas las posibilidades que permite la herramienta. A su vez, poder compartir proyectos facilita el uso de la herramienta porque uno puede ayudarse con el trabajo ya realizado por otros y puede formar parte de una comunidad global que comparte el interés de aprender junto con otros sobre un tema común que los motiva.

Si bien en esta actividad es de esperar que muchos estudiantes aún no estén familiarizados con el lenguaje de programación de Scratch, para futuros encuentros lo interesante de esta sección es que quien la visita no solo puede acceder al producto realizado por el "scratcher" sino que también puede acceder a la programación en la sección "Ver dentro". De esta manera, uno puede aprender de lo que otros ya han hecho, y a su vez, puede realizar modificaciones sobre esos proyectos, explorando y creando nuevas alternativas para ellos.

Nombre actividad	Explorando la interfaz de Scratch
Objetivo	Explorar las posibilidades de la interfaz de Scratch. Fomentar la búsqueda autónoma de la información.
Áreas de conocimiento	Ciencias de la computación
Conceptos relacionados	Manejo interfaz (objetos y escenario)
Competencias	Programación. Investigación y experimentación
Materiales	Terminal, conectividad y proyector.
Enlaces de interés	https://www.youtube.com/watch?v=2xMQ3qsFIsM
Anexo	

Desarrollo de la actividad **Explorando la interfaz de Scratch**

El propósito de esta actividad es que los estudiantes puedan conocer la interfaz de manera autónoma, explorándola libremente, para que cada cual pueda recorrerla siguiendo su propio camino.
Luego de esta primera indagación libre de la interfaz, se les proponen a los estudiantes diferentes desafíos para orientar una búsqueda más precisa de la información. A modo de ejemplo:



- Añadir objetos de diferentes maneras (seleccionar desde la biblioteca, dibujar, cargar desde archivo, sacar foto).

- Añadir un sonido diferente para cada objeto (seleccionar desde la biblioteca y crear sonidos propios).

- Cambiar el escenario variando el origen (biblioteca, dibujo, archivo, foto).

Es interesante que, a lo largo de esta exploración, los participantes puedan ayudarse y compartir unos con otros todo lo que van descubriendo, para promover ante todo la colaboración entre los participantes.

Accediendo a este link https://www.youtube.com/watch?v=2xMQ3qsFIsM se puede encontrar un video tutorial que guía a quienes se están iniciando con la herramienta para poder realizar estos primeros pasos.

Nombre actividad	Iniciándose en la programación con Scratch
Objetivo	Introducirse en el pseudocódigo de Scratch. Elaborar algoritmos simples de programación.
Áreas de conocimiento	Ciencias de la computación
Conceptos relacionados	Secuencia, Algoritmo, programación de bloques.
Competencias	Resolución de problemas. Programación. Investigación y experimentación
Materiales	Terminal y conectividad
Enlaces de interés	Link Video Tutorial: https://www.youtube.com/watch?v=LE9tKuntaFI Cartas scratch: https://scratch.mit.edu/info/cards/
Anexo	

Desarrollo de la actividad **Iniciándose en la programación con Scratch**

El propósito de esta actividad es familiarizarse con el pseudocódigo de Scratch, basado en la programación de bloques.

Se propone a los participantes que añadan un objeto de la biblioteca y que exploren todas las posibilidades que se abren en la pestaña Programas, como por ejemplo: Movimiento, Apariencia, Sonido, Eventos, Control. Para ejercitar el uso de estos bloques, los participantes pueden otorgarle movimiento a sus objetos utilizando distintos tipos de algoritmos simples.

Se espera que los estudiantes vayan haciendo su propio recorrido en el programa, de manera de brindarles ayuda y orientación de acuerdo a sus necesidades. Es importante que al final de esta actividad cada participante pueda mostrar a los demás cómo lo hizo para aportar al trabajo colaborativo.

Accediendo a este video tutorial https://www.youtube.com/watch?v=LE9tKuntaFI se podrá encontrar una guía para la realización de esta actividad, continuando el proyecto realizado en el primer tutorial.

En los siguientes encuentros se puede profundizar en la utilización de la herramienta a través de las cartas de Scratch (ver link de interés). Las cartas contienen una serie de ejercicios y sus resoluciones, que trabajan los conceptos principales de la programación de bloques.

Nombre actividad	Creando mi primer Videojuego en Scratch
Objetivo	Introducirse en la creación de un videojuego. Profundizar en la programación de bloque**s**
Áreas de conocimiento	Matemática. Ciencias de la computación
Conceptos relacionados	Secuencia, Algoritmo, estructuras de control, variables.
Competencias	Resolución de problemas. Proyección. Gestión de proyectos. Programación. Trabajo en equipo. Diseño y creatividad. Comunicación. Investigación y experimentación
Materiales	Terminal, conectividad y proyector.
Enlaces de interés	Video tutorial 3: https://www.youtube.com/watch?v=mCu77luLJXo Video tutorial 4: https://www.youtube.com/watch?v=qKlN6kz2QQs Video tutorial 5:https://www.youtube.com/watch?v=wn82nUJdZRk
Anexo	

Desarrollo de la actividad **Creando mi primer Videojuego en Scratch**

Esta actividad puede suponer varios encuentros ya que el proceso de crear un videojuego implica numerosos pasos a ser realizados. Los 3 videos tutoriales funcionan como guía para comenzar la creación de diferentes videojuegos. Aunque en este caso se realiza una carrera de autos, la programación puede ser utilizada y mejorada para cualquier juego.

Se les propone a los niños comenzar a programar un videojuego modelo de carrera de autos que sirva como disparador para introducirse en la complejidad de programar.

Es interesante discutir con los participantes todo lo que implica para ellos la construcción de un videojuego de una carrera de autos para poder visualizar el problema general y luego descomponerlo en problemas más pequeños. De esta manera los participantes se introducen en la lógica "divide y vencerás", comúnmente conocida en las ciencias de la computación. Este proceso consiste en dividir un problema en muchos subproblemas de manera de facilitar su resolución.

En los sucesivos encuentros los participantes podrán resolver los siguientes problemas: diseño de la pista de autos y de los autos de la carrera; diseño de las pantallas del comienzo y del final del juego; programación del movimiento de los autos; programación de los sonidos del videojuego; construcción de variables para añadirle vidas, energía, vueltas, entre otros problemas que pueden surgir de acuerdo a la creatividad que tengan a la hora de programar.

Es importante tener en cuenta los intereses y las capacidades de los participantes a la hora de programar un videojuego en equipo, ya que teniendo en cuenta esas variables podrán elegir en qué parte del proceso de creación quieren intervenir.

5.2. Robótica educativa con kits Lego

La robótica educativa es una potente herramienta para trabajar en el aula ya que permite desarrollar habilidades y trabajar distintos conceptos a partir de una temática que motiva y atrae a los estudiantes.

Además de ser motivadora, la robótica educativa permite trabajar en base a problemas y a sus posibles soluciones, conjugando el diseño, la construcción y la programación de un dispositivo, y poniendo en juego tanto el trabajo manual como el intelectual. A su vez, integra diversas disciplinas, en las que cada estudiante puede aportar en función de sus capacidades e intereses, sin perder de vista la visión sistémica del problema a resolver. La resolución de un problema en robótica es el resultado de un gran trabajo en equipo que debe procurar relacionar y ensamblar el aporte de todos.

Así como la robótica favorece el trabajo en equipo, también fomenta el desarrollo de habilidades socioemocionales como la comunicación asertiva, la perseverancia y la tolerancia a la frustración.

Aportes para el uso responsable

Trabajar con robótica educativa implica manipular una cantidad de piezas, a veces muy pequeñas y llamativas. Es necesario que los docentes no teman usar los kits por miedo a que se pierdan o se deterioren las piezas; a la larga, el propio uso de cualquier kit tiene como consecuencias asociadas su pérdida y deterioro.

Pero el docente sí puede contribuir a desarrollar en el grupo una cultura de cuidado de los kits. Para ello es imprescindible acordar previamente con los estudiantes ciertos puntos que garanticen un buen uso y un buen mantenimiento del recurso. Con esto se pretende que sean conscientes que un mismo kit puede ser utilizado por muchas personas y que por ende se requiere que todos ayuden para que las piezas no se pierdan y se mantenga cierto orden. Es bueno destinar unos cinco o diez minutos finales de la actividad para el desarmado y la organización de las piezas que fueron utilizadas. También es recomendable que cuando se trabaja con un kit determinado, el primer acercamiento a él implique la visualización de todo lo que contiene, así como una libre experimentación de sus piezas, de manera que los estudiantes puedan conocer y familiarizarse intuitivamente con el curso.

A continuación, a modo de ejemplo, se profundizará en el trabajo con dos posibles kits: *Lego WeDO* y *Lego Mindstorms*. Finalmente, se desarrollarán las potencialidades de la participación en competencias de deportes robóticos.

5.2.1. Lego WeDo

El kit de construcción básico de *Lego WeDo* contiene más de 150 piezas, motores y sensores de movimiento y de inclinación, que permiten construir y programar prototipos de diversa complejidad.

Además, contiene una placa llamada Hub USB Lego que permite conectar el robot con la computadora. El robot puede ser programado tanto con el software *Lego Education WeDo* como con Scratch.

En el sitio web http://ro-botica.com/ hay disponible una guía rápida para el profesor así como instrucciones de montaje de 12 modelos robóticos para construir.

Nombre actividad	Discutiendo sobre qué es un robot
Objetivo	Introducirse en la temática de la robótica. Discutir colectivamente sobre las características principales de un robot.
Áreas de conocimiento	Ciencias de la computación. Lengua
Conceptos relacionados	Robot
Competencias	Comunicación
Materiales	Hojas, lápices, computadoras, conectividad
Enlaces de interés	
Anexo	

Desarrollo de la actividad **Discutiendo sobre qué es un robot**

El propósito de esta actividad es reflexionar colectivamente sobre lo que es un robot a partir de las ideas previas que los participantes tienen al respecto. Se irá conectando la discusión con su vida cotidiana.

Un posible disparador para la actividad es proponer a los estudiantes que dibujen un robot, para luego en base a lo dibujado discutir sus características.

Es importante intercambiar opiniones en la relación entre los robots y la necesidad de solucionar problemas por parte del hombre. También ayuda para la discusión reflexionar sobre los distintos robots que nos rodean a diario, y discutir sobre si todos cumplen los requisitos para ser un robot o no. A modo de ejemplo, muchas veces surge en este tipo de discusiones la pregunta si todos los electrodomésticos son robots, y en caso contrario, ¿por qué no?
Es conveniente que al finalizar la discusión se puedan pasar en limpio las características principales de todo robot, como, por ejemplo: que es un dispositivo programable, posee sensores que lo relacionan con el medio, que posee actuadores, cumple alguna función, entre otras. Para terminar la actividad, se puede proponer a los estudiantes que hagan una búsqueda en YouTube sobre diferentes tipos de robots, que luego pueden ser compartidos y observados colectivamente de manera de visualizar la amplísima variedad existente.

Nombre actividad	Construyendo un robot siguiendo instrucciones
Objetivo	Lograr la construcción colectiva de un robot a partir de una secuencia de instrucciones.
Áreas de conocimiento	Mecánica
Conceptos relacionados	Instrucción, Secuencia.
Competencias	Trabajo en equipo
Materiales	Kit de Lego We Do, terminal y conectividad.
Enlaces de interés	Guía rápida para el profesor: http://manager.ro-botica.com/uploads/items/ITEM_4730_DOCPROD.pdf Instrucciones de montaje: http://www.ro-botica.com/tienda/LEGO-Education/LEGO-WeDo/
Anexo	

Desarrollo de la actividad Construyendo un robot siguiendo instrucciones

En una primera instancia se explora el kit colectivamente, visualizando las diferentes piezas y las funciones de las piezas claves (motores, sensor de movimiento y de inclinación, placa Hub USB).

Luego, en equipo, los participantes deberán construir un robot siguiendo una serie de instrucciones para armarlo. Es conveniente que los equipos no sean muy numerosos para que todos los integrantes puedan participar activamente del armado.

Es importante que antes de comenzar el armado se colectivice sobre cómo seguir las instrucciones, para asegurarse que todos comprendieron la lógica del instructivo (primero se presentan las piezas necesarias y luego se explica cómo se ensamblan).

En el link http://www.ro-botica.com/tienda/LEGO-Education/LEGO-WeDo/ se puede acceder a instrucciones de montaje para 12 modelos diferentes.

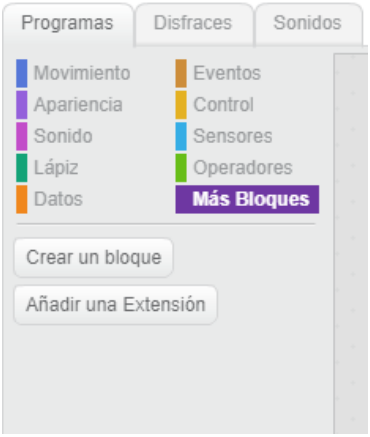
Instrucciones de montaje de 12 modelos, en PDF					
 Pájaro volador	 Mono percusionista	 Pájaros danzarines	 Barco navegante	 Gigante colgante	 Afición ruidosa
 Cocodrilo hambriento	 León rugiente	 Chutador a gol	 Portero parapelotas	 Avión de rescate	 Hilador inteligente

Nombre actividad	Programando mi robot
Objetivo	Introducirse en la programación y la codificación de instrucciones adaptándose a un lenguaje.
Áreas de conocimiento	Ciencias de la computación
Conceptos relacionados	Lenguaje de programación, instrucción, secuencia, algoritmo, estructura de control.
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Investigación y experimentación
Materiales	Kit de Lego We Do, terminal y conectividad.
Enlaces de interés	Guía rápida para el profesor: http://manager.ro-botica.com/uploads/items/ITEM_4730_DOCPROD.pdf
Anexo	

Desarrollo de la actividad **Programando mi robot**

Para programar un robot construido con el kit de *Lego WeDo* se puede utilizar tanto el software *LEGO Education WeDo* como Scratch. Optamos por este último ya que profundiza el trabajo realizado en la unidad de programación.

Una vez armado el robot se debe conectar la placa Hub USB con una computadora y luego iniciar la sesión en la plataforma de Scratch online. Ya en la plataforma, se debe ingresar en la opción “Crear”. De esta forma, aparecerán los bloques de programación. Para manejar el *Lego WeDo* desde la PC se deben realizar los siguientes pasos:



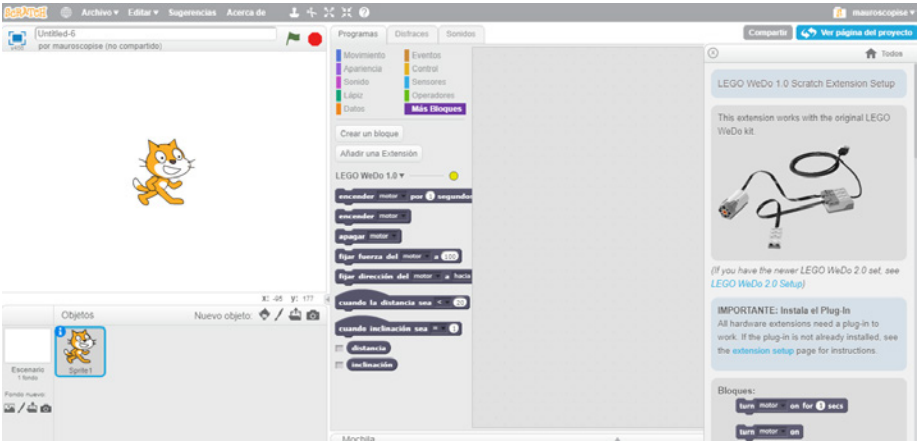
1) Ir a la opción “Más Bloques”

2) Ir a “Añadir una Extensión” y elegir la opción “Lego WeDo 1.0”.

3) Luego de agregada la extensión aparecen varios bloques nuevos a las opciones de programación.

4) Podemos verificar la conexión entre el robot y *Scratch*. Si este botón está verde, la conexión se estableció correctamente.
Si está rojo, se debe hacer clic sobre él y se despliega la ventana de extensiones.

5) Debemos instalar la extensión dependiendo del navegador que utilizamos.



A continuación, a modo de ejemplo, se trabajará la programación en Scratch del “Chutador a gol” y del “Portero parapelotas”, ambos diseños que figuran en la lista de instrucciones de montaje. Lo entretenido de estos dos modelos es que una vez programados se puede jugar con ambos a la vez, por ejemplo a los penales.

Para aquellos participantes que ya conocen Scratch, es interesante proponerles diferentes desafíos que les permitan familiarizarse con los nuevos bloques. Por ejemplo: buscar algoritmos que permitan al “Chutador” golpear una pequeña pelota de papel. Lo interesante es que los participantes vayan probando sus propios algoritmos y que por su ejecución ellos mismos puedan ir depurando los errores encontrados y perfeccionando su algoritmo.

A modo de ejemplo, se presentará un sencillo programa.

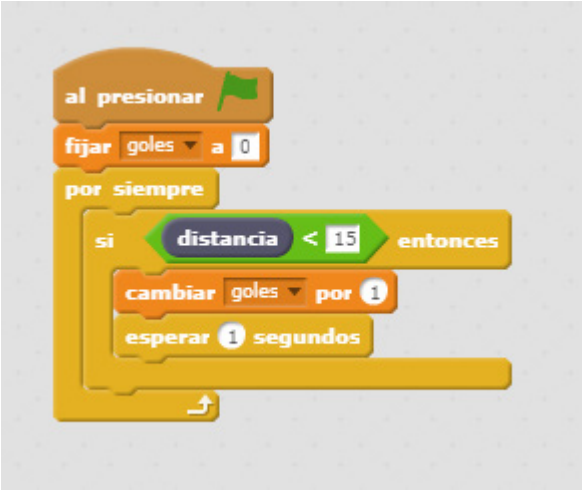
En este ejemplo el chutador pateará cada vez que se presione la tecla espacio. Se debe variar el tiempo de encendido del motor para que se asemeje a un jugador pateando y no gire muchas vueltas.

Nombre actividad	Robot con sensor
Objetivo	Introducirse en la programación de un sensor.
Áreas de conocimiento	Mecánica. Ciencias de la computación. Ciencias físicas
Conceptos relacionados	Sensor
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Investigación y experimentación
Materiales	Kit de Lego We Do, terminal y conectividad.
Enlaces de interés	
Anexo	

Desarrollo de la actividad **Robot con sensor de distancia**

En esta instancia se construye un robot en el cual se incluye un sensor de distancia y se trabaja la programación. Los modelos que cuentan con sensor de distancia son: “Cocodrilo hambriento”, “Portero Parapelotas”, “Afición Ruidosa” e “Hilador Inteligente”. También se puede agregar el sensor a los modelos “Pájaro Volador” y “Gigante colgando”.

Una vez armado, se puede explorar cómo funciona el sensor. Para ello se utilizan los bloques del módulo "Sensores" y se programa de forma que puedan visualizarse los valores que mide el sensor en la pantalla. El personaje de Scratch desplegará un globo de historieta que indica los diferentes valores que mide el sensor de distancia.



Continuando con el ejemplo del "Portero Parapelotas", se puede programar el sensor de este para que cumpla diferentes funciones. Por ejemplo: buscar algoritmos que permitan a través del sensor contabilizar la cantidad de goles. Al igual que en la actividad anterior los participantes pueden ir probando sus propios algoritmos y en base a su ejecución, pueden ir depurando los errores encontrados y perfeccionando el funcionamiento. Este algoritmo puede utilizarse para contabilizar los tantos que se le hacen al golero. Mediante una variable denominada "Goles", se sumará uno cada vez que un objeto pase a menos de 15 centímetros del sensor.

5.2.2. Lego Mindstorms

Lego Mindstorms NXT es un kit de robótica programable realizado por Lego en el 2006. Existen diferentes versiones del mismo. Las últimas versiones son Lego Mindstorms NXT 2.0 y EV3.

El kit cuenta con un microcontrolador, bloque de NXT o ladrillo. En él existen cuatro entradas para los sensores y tres salidas de energía, localizadas en la parte posterior del bloque, haciendo que la conexión para los motores y partes móviles sea de fácil

acceso. El ladrillo de NXT puede comunicarse con la computadora mediante una entrada USB. Además, para comunicarse con otros robots en las cercanías puede utilizarse Bluetooth. Esta conectividad con Bluetooth no solo permite conectarse con otros ladrillos de NXT, sino también con computadoras, teléfonos móviles, tablets y otros dispositivos.
En el sitio web http://www.nxtprograms.com están disponibles instrucciones de montaje de diferentes modelos robóticos.

Existen diferentes entornos de programación que pueden utilizarse para *Lego Mindstorms NXT.* Uno utilizado para facilitar el acceso de niños, niñas y adolescentes a la programación es el *Enchanting.* Este es un software de programación visual basado en *Scratch.* El plan Ceibal elaboró un documento que guía la instalación y la utilización de *Enchanting*. Se encuentra disponible en el siguiente enlace: http://www.santiagoavila.com/Libros/Programacion_Enchanting.pdf.

Nombre actividad	Armado y programación de robot
Objetivo	Introducirse en la robótica. Lograr la construcción colectiva de un robot a partir de una secuencia de instrucciones. Aproximación a la función y programación del robot armado. Generar un espacio de trabajo colectivo y colaborativo.
Áreas de conocimiento	Ciencias de la computación. Mecánica
Conceptos relacionados	Secuencia, lógica.
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Diseño y creatividad
Materiales	Kit NXT, computadoras.
Enlaces de interés	Instrucciones de montajes disponibles en: http://www.nxtprograms.com/ Descarga del entorno de programación: http://enchanting.robotclub.ab.ca/tiki-index.php
Anexo	

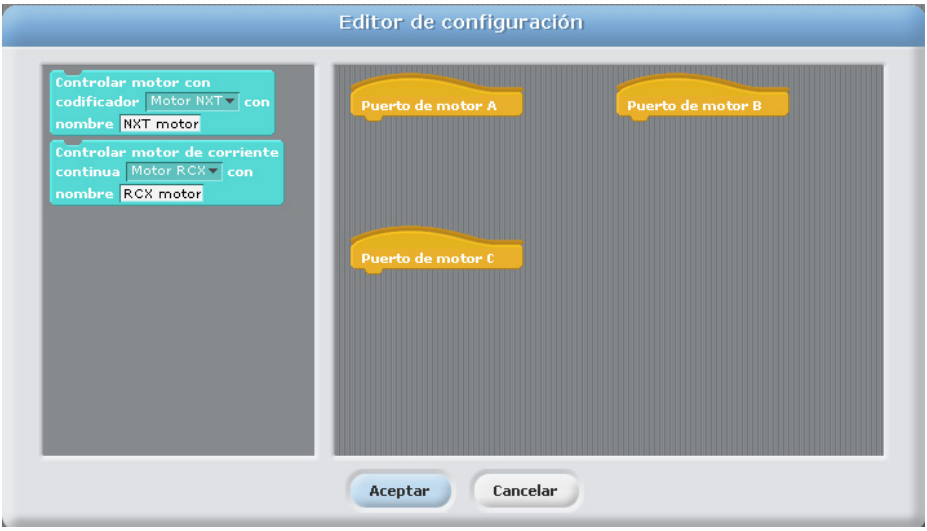
Desarrollo de la actividad **Armado y programación de robot**

En una primera instancia se trabaja en el reconocimiento del kit. Se exploran las diferentes piezas y sus funciones. Se trabaja el concepto de motor y de sensor.

Se buscan diferentes prototipos de robot en la página nxtprogram.com. Se elige alguno de los modelos para armar de base. Para las versiones *Lego Mindstorms NXT 1.0* o *Lego Mindstorms NXT 2.0* se pueden utilizar para una primera aproximación los modelos: "*Castor Bot*", "*Motor Chassis*" o "*Five Minute Bot*".

Posteriormente se comienza con la programación en *Enchanting*. Primero se explora la interfaz. Cada vez que abrimos *Enchanting* lo primero que hay que hacer es indicar qué sensores y motores tenemos en el bloque NXT y su ubicación. Para configurar los motores se presiona "Configurar Motores" dentro de la pestaña de Motores.

Aparecerá una ventana en la que seleccionamos los motores, las posiciones en la que están conectados y el nombre que deseamos darle. Solo es necesario conectar el bloque azul "Controlar motor con codificador" con el bloque que corresponda al puerto que tengamos en uso. Es decir, se asocia el motor al puerto al cual está conectado en el controlador NXT (puertos A, B y C) y se le puede poner un nombre, por ejemplo: "izquierdo".

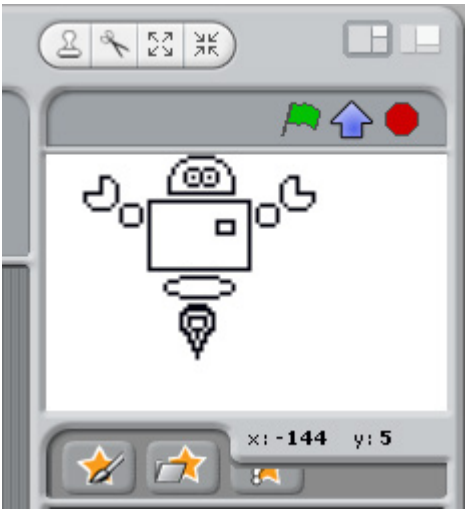


Al asociar los Motores a los Puertos aparecerán nuevos bloques con los que podemos programar los motores.

Se deja que en un principio los participantes exploren libremente los diferentes bloques que aparecen, cuestionando para qué será uno u otro. A continuación se plantean diferentes desafíos para lograr el movimiento del robot:

- que logre avanzar
- que logre retroceder
- que logre girar
- que forme un cuadrado.

Los programas se irán cargando en el ladrillo NXT y probando de forma que puedan ajustarse hasta lograr los diferentes objetivos. Para descargar el programa en el ladrillo NXT hay que pulsar en la flecha azul que hay entre los botones de "inicio" (bandera verde) y "pare" usuales de Scratch, situados en la zona superior derecha de la pantalla. Mientras no tengamos conectado y encendido el robot las tres opciones no estarán disponibles, apareciendo todas con el símbolo de prohibido.

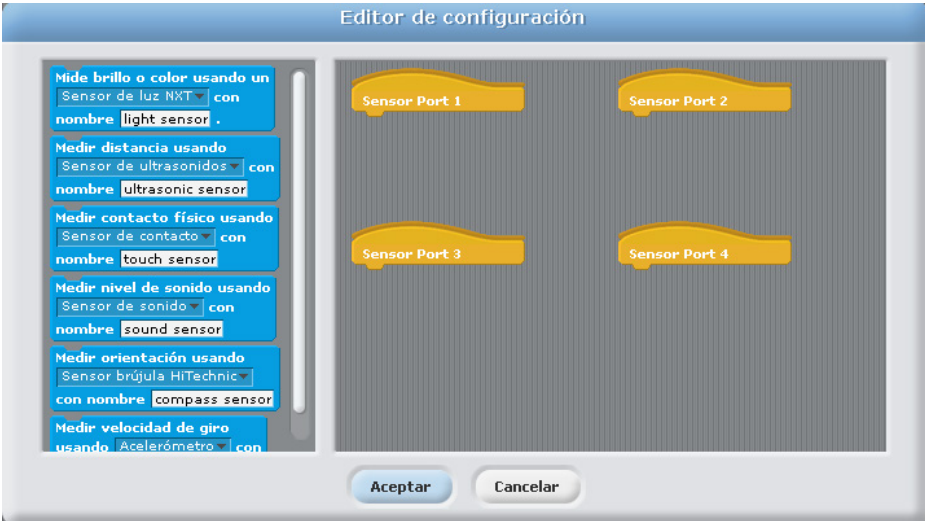


Nombre actividad	Sensores y actuadores
Objetivo	Acercamiento al concepto de sensor. Construir un dispositivo con sensores. Lograr la resolución de problemas utilizando la programación y fomentando la investigación.
Áreas de conocimiento	Matemática. Ciencias de la computación. Mecánica
Conceptos relacionados	Sensores, programación, secuencia, lògica.
Competencias	Resolución de problemas. Programación. Trabajo en equipo .Diseño y creatividad. Investigación y experimentación
Materiales	Kit NXT, computadoras
Enlaces de interés	http://www.nxtprograms.com/
Anexo	

Desarrollo de la actividad **Sensores y actuadores**

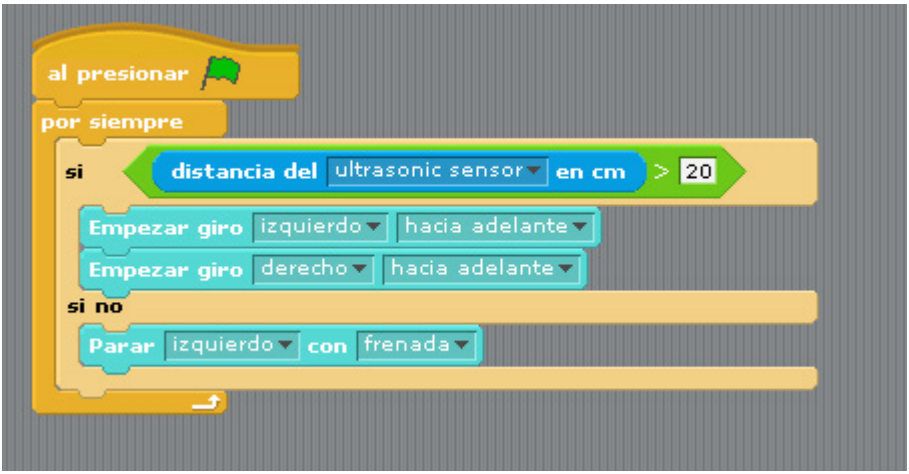
Se visualizan los diferentes sensores que trae el kit y sus funciones. El kit cuenta con 4 tipos de sensores: de luz, de distancia, de contacto y de sonido. Se parte de un modelo armado y se plantea incorporar los sensores para cumplir determinadas funciones. Se deja que los participantes exploren y prueben diferentes formas de colocarlos.

Una vez incorporados los sensores en el modelo se plantean desafíos que deberán resolverse a través de la programación del robot. Es necesario configurar los sensores en el *Enchanting* para programarlos. Para ello dentro del bloque “Sensores” se presiona “Configurar Sensores”. Allí aparecerán los bloques de cada tipo de sensor. Para poder utilizarlos se debe asociar el sensor al puerto al cual está conectado en el ladrillo NXT (puertos 1, 2,3 o 4). Al asociar Sensores a los Puertos aparecerán nuevos bloques para cada tipo de sensor.



Los participantes pueden ir probando diferentes alternativas para resolver los desafíos. En una primera instancia pueden plantearse desafíos para cada sensor: que el robot logre moverse sin chocar con ningún objeto utilizando el sensor de distancia, que consiga avanzar cuando el piso sea de determinado color utilizando el sensor de luz, que siga una línea de determinado color en el piso, etc.

Un posible algoritmo para un robot que esquive objetos es el siguiente:



Este simple algoritmo, que puede ser mejorado, realiza un procedimiento para esquivar obstáculos. Lo que hará el robot es básico: encenderá los dos motores siempre que la distancia del sensor sea mayor a 20 cm y se mantendrán encendidos hasta que la distancia sea menor. Cuando esto suceda, apagará el motor izquierdo provocando un giro hacia la izquierda, teniendo cuenta que el motor derecho continuará encendido.

5.2.3. Deportes Robóticos

Una de las posibilidades para trabajar todas las temáticas abordadas de una forma motivadora y generando objetivos concretos con los niños, niñas y adolescentes es la participación en competencias como el Sumo o las Olimpiadas Ceibal.

En distintas instituciones de enseñanza e investigación a nivel mundial se desarrollan torneos deportivos donde los jugadores son robots autónomos. Uno de los deportes que se toma como base es el Sumo, el cual es un tipo de lucha donde dos contrincantes se enfrentan a un duelo en un área circular llamada Dojo. Hoy en día se ha diversificado la propuesta existiendo diferentes tipos de competencias robóticas.
En cuanto al Sumo, se gana cuando uno de los luchadores obliga al oponente a que ponga el hombro en el suelo o consigue echarlo fuera del círculo. De igual forma, en el Sumo robótico el objetivo es expulsar al contrincante del Dojo.
En el año 2004 el Instituto de Computación de Facultad de Ingeniería de la Universidad de la República organizó el Primer Campeonato Uruguayo de Sumo Robótico. A partir de entonces, este evento se viene desarrollando año a año con un número creciente de participantes que disfrutan, ya sea como competidores o como observadores, de una actividad abierta a todo público.

Los robots que compiten en las diferentes categorías deben ser dispositivos móviles completamente autónomos. Es decir, deben ser capaces de desplazarse a través de los escenarios y cumplir los objetivos planteados sin intervención de ningún tipo. Es por este motivo que crear y programar un robot para estas competencias es un gran desafío que implica un importante trabajo en equipo y la resolución de problemas que van apareciendo constantemente.

Para más información sobre la competencia de sumo que se realiza en Montevideo se puede ingresar en https://sumo.uy.

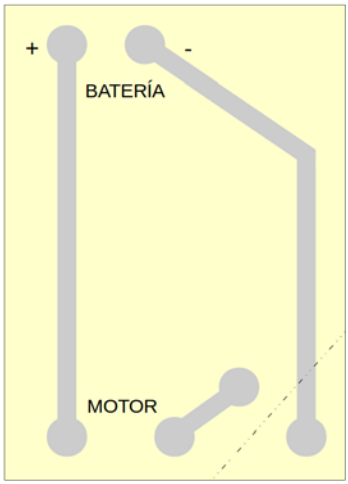
5.3. Electrónica

5.3.1. Electrónica analógica.

Actividades con electrónica analógica:

Nombre actividad	Circuitos con Papel
Objetivo	Trabajar sobre cómo funcionan los circuitos eléctricos y generalidades sobre la electricidad.
Áreas de conocimiento	Ciencias físicas. Electrónica.
Conceptos relacionados	Corriente alterna/continua, material conductor y no conductor, concepto de cortocircuito.
Competencias	Trabajo en equipo. Investigación y experimentación.
Materiales	Hoja, papel de aluminio, pegamento, motor eléctrico pequeño, cables cocodrilo, batería de 9v.
Enlaces de interés	
Anexo	

A partir de la pregunta ¿qué es un robot? los participantes deben imaginar cuáles son sus componentes y se elabora una lista con las cosas que se proponen. A partir de la diferenciación entre hardware y software se reordenan los componentes que fueron propuestos.



Nos detenemos en los conceptos o ideas que sean más cercanos a la energía, para a partir de esto introducirnos en la electrónica que nos ayudará a poner en funcionamiento el robot. Brevemente presentamos cómo funcionan los circuitos eléctricos así como algunas generalidades sobre la electricidad:

- Materiales conductores y no conductores
- Distintos tipos de corriente (alterna y continua)
- Concepto de cortocircuito

Una vez presentadas y enriquecidas las ideas generales por los aportes y preguntas de los estudiantes, nos disponemos a armar nuestro primer circuito eléctrico.

Dibujamos un circuito en el pizarrón, los estudiantes lo copian en una hoja y luego lo repasan con papel aluminio. El esquema debe quedar dibujado de forma tal que al doblar la esquina inferior derecha de la hoja estemos cerrando el circuito. La esquina de la hoja se convierte en un interruptor; si doblamos y presionamos, el circuito se cierra y los motores que le conectamos se encienden; y si no la doblamos, el circuito queda abierto y los motores permanecen apagados.

Una vez armado el circuito con papel aluminio en las pistas conductoras es el momento de conectarles, con ayuda de unos cables cocodrilo, un pequeño motor eléctrico y una batería de 9v en los terminales que quedaron en uno y otro extremo del circuito.

Luego de armado el circuito y repetido el ejercicio de doblar y desdoblar la esquina de la hoja para hacer que el motor se encienda y apague, es hora de volver a la teoría: ¿Pueden los participantes explicar cómo funciona el circuito que hemos construido? ¿Cómo podemos hacer para que el motor gire hacia el otro lado?

Nombre actividad	Motor eléctrico casero
Objetivo	Vincularnos en forma práctica con los conceptos básicos detrás de los actuadores (dispositivos que convierten la electricidad en trabajo sobre el mundo físico <movimiento, luz, calor, sonido, ondas electromagnéticas, entre otras>). Entender el funcionamiento de un motor eléctrico.
Áreas de conocimiento	Ciencias físicas. Electrónica.
Conceptos relacionados	Corriente continua, materiales conductores y no conductores, cortocircuito.
Competencias	Trabajo en equipo. Investigación y experimentación.
Materiales	Por subgrupo: destornilladores de precisión (diferentes tamaños y formas, incluyendo tipo estrella), 1 disco duro para destruir, 2 pilas medianas 1.5V, alambre de cobre barnizado para bobinas (1-2mm), cinta aisladora, 2 clips, tijera.
Enlaces de interés	
Anexo	

Una buena manera de comenzar con esta actividad es proponer una discusión grupal en torno a la pregunta: ¿De dónde obtienen el movimiento los robots?
Si el grupo ya ha realizado una actividad como la anterior, puede partir por recordar las conclusiones que sacamos al respecto de los materiales conductores y no-conductores, los circuitos eléctricos y los motores. Luego de esta charla introductoria propondremos una actividad manual donde construiremos nuestro propio motor eléctrico.

Se sugiere que, en subgrupos (máximo 4 participantes), desarmen un disco duro de computadora usando las herramientas adecuadas y observen las piezas que contiene, para discutir sobre qué son y qué función cumplen. Les pediremos que separen uno de los potentes imanes que hay dentro del disco duro.

Utilizando estos imanes construiremos en cada subgrupo, siguiendo instrucciones del educador, un motor eléctrico sencillo en base a los materiales proporcionados. Las instrucciones para construir un motor como este: https://www.youtube.com/watch?v=kUjS0aJxQjk.

Luego de armado y puesto en marcha, intentaremos explicar entre todos su funcionamiento.

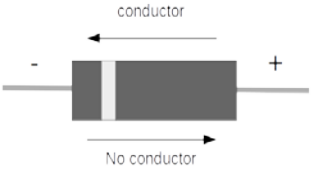
Nombre actividad	Diseño de circuitos con la protoboard
Objetivo	Familiarizarnos con el uso de la protoboard. Reconocer distintos componentes y diseñar circuitos eléctricos.
Áreas de conocimiento	Electrónica
Conceptos relacionados	Corriente continua, resistencia eléctrica, circuitos en serie y paralelo, protoboard, cortocircuito.
Competencias	Resolución de problemas. Trabajo en equipo. Investigación y experimentación
Materiales	Protoboard, luces leds, resistencias (470Ω), interruptores, potenciómetros, baterías 9v.
Enlaces de interés	
Anexo	

Esta actividad puede plantearse como una continuación de las anteriores, ya que seguiremos profundizando en los circuitos eléctricos. Por otro lado, puede ser un punto de partida para las actividades de electrónica digital que propondremos más adelante y que se basan en el uso de la *protoboard* para el prototipado de circuitos eléctricos. La placa de prototipado, protoplaca o *protoboard* es una herramienta que permite conectar y desconectar fácilmente cables y componentes para evitar tener que usar soldaduras.

Las propuestas para este tema pueden ser variadas, en función de la profundidad que se quiera lograr en el conocimiento de la electrónica. El camino que aquí seguimos se adapta al objetivo concreto de familiarizarnos con el uso de la protoboard y la construcción de circuitos en serie y en paralelo. El producto será un circuito que permite encender unas luces LED usando interruptores y potenciómetros.

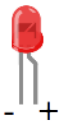
Dependiendo de la disponibilidad material y la proyección en que se enmarque esta actividad en el proceso grupal, puede ser interesante introducir aquí el uso del multímetro, el cual permite medir continuidad, resistencias y voltajes.

Si no se ha trabajado antes con circuitos con luces LED, es buen momento para conversar un poco sobre este componente electrónico. Los LED (por su sigla en inglés) son Diodos Emisores de Luz, que como los demás diodos, son componentes polarizados que actúan como conductores de la electricidad desde un polo positivo a uno negativo y como no-conductores si aplicamos la corriente inversa.



En este diodo la línea blanca muestra el lado negativo. Podemos probarlo con la función de medición de diodos del multímetro, que reconocemos por el símbolo universal para los diodos:  . Si respetamos la polaridad del diodo al conectarle los electrodos del multímetro, este nos devolverá un sonido "beep" y pondrá un valor cercano a 0 en la pantalla. Si en cambio lo conectamos con la polaridad invertida el multímetro mostrará un número 1 en la pantalla, que significa conductividad nula.

En el caso de los LED, el polo negativo puede reconocerse gracias a que suele tener la terminal de conexión más corta que el positivo:



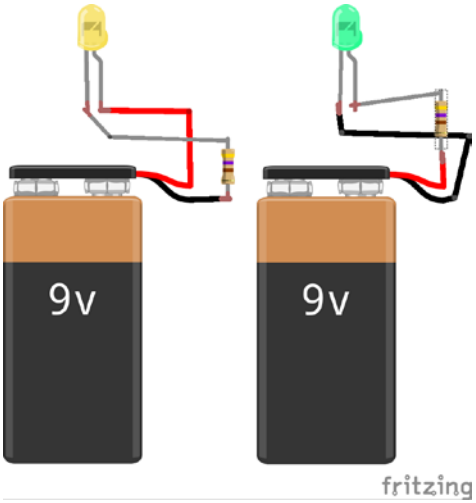
Otro componente que debemos reconocer para construir un circuito con luces LED es la resistencia eléctrica.

Podemos comenzar por plantear el problema: los LED consumen muy poca energía. Si conectamos un LED directamente a los polos de la batería, este brillaría muy intensamente durante una fracción de segundo y se quemaría. ¿Qué podemos hacer para evitarlo?
Dos buenas respuestas que nos han dado los niños:

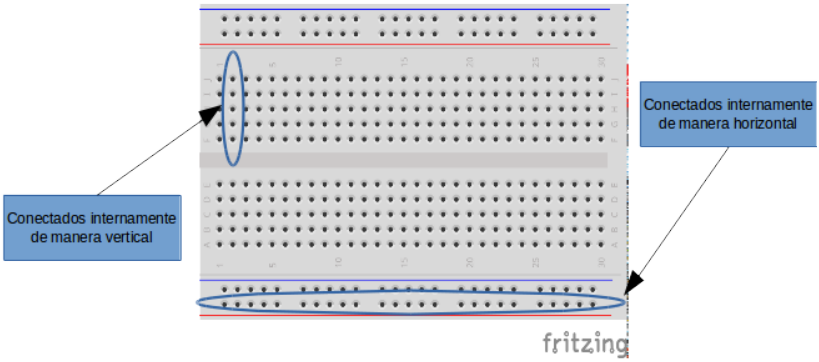
- Usemos una batería más chica (menos potente).
- Pongamos otros LED en el circuito para que consuman el excedente.

Como no tenemos una fuente más pequeña (solo tenemos una batería de 9v) recurriremos a una solución similar a la de la segunda respuesta. Solo queremos encender un LED, por lo que usaremos un componente que también es capaz de consumir parte de la corriente, sin emitir luz: la resistencia.

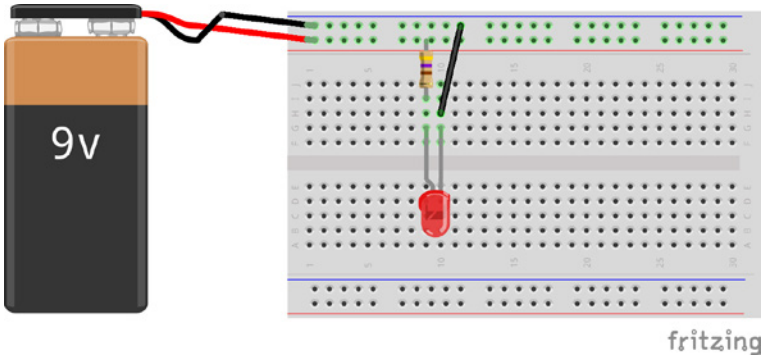
Los niños podrían tener algunas ideas previas respecto a este componente, sobre todo por su uso como emisores de calor. Si no fuese el caso, podemos proponer una adivinanza: ¿Qué sucede con la energía que la resistencia no deja pasar?
Luego de conversar la idea de las resistencias eléctricas, pasemos a construir un circuito sencillo que encienda un LED.
Tal vez alguno de los participantes se anime a dibujar el circuito en el pizarrón. También podemos dibujar los componentes y preguntar cómo los conectarían. Hay varias formas correctas:



Lo importante es respetar la polaridad de los LED. Las resistencias en cambio no tienen polaridad y pueden ser conectadas en cualquier sentido. Además cumplen su función sin importar en qué lugar del circuito las coloquemos, antes o después del actuador (el LED).
Es hora de construir un circuito real con la *protoboard.*

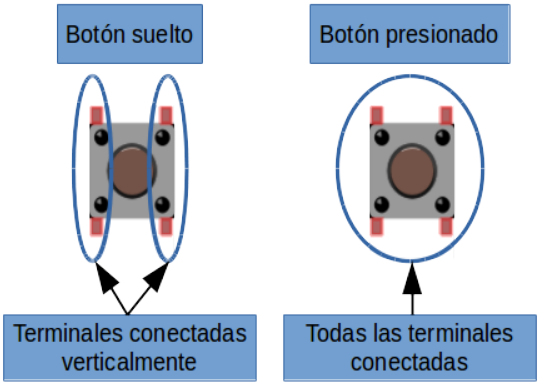


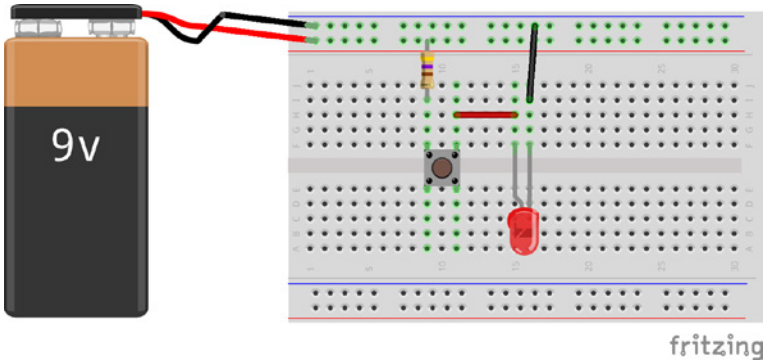
Podemos dibujar la protoboard y la batería en el pizarrón y guiar a los participantes para que dibujen el circuito incluyendo las resistencias y los LED.



Una vez presentado el recorrido de la corriente en el dibujo, los estudiantes pasarán a construir lo aprendido mediante un circuito eléctrico.

Paso a paso, podemos ir aumentando la complejidad de nuestros circuitos. Añadiremos un interruptor de tipo pulsador. Este componente funciona como conductor entre las terminales de la derecha y la izquierda solo cuando presionamos el botón.

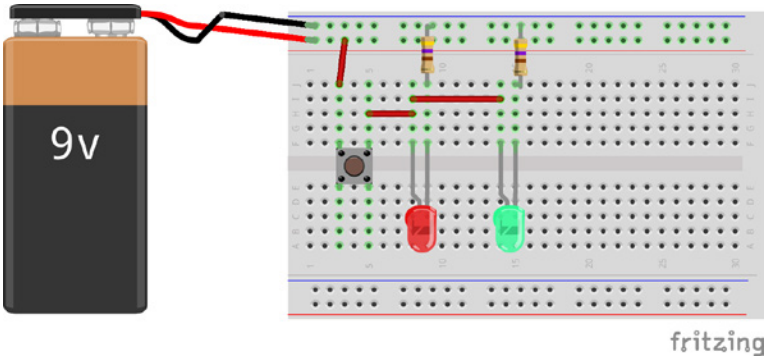




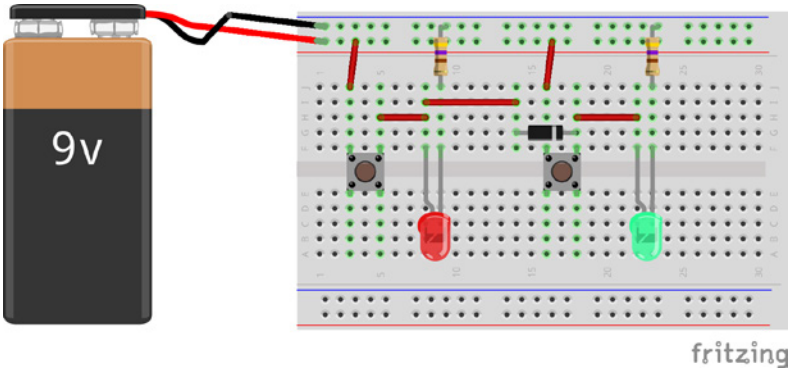
Se puede profundizar agregando desafíos:
¿Cómo conectarían otro LED para que también se encienda cuando presionamos el pulsador?

Usando dos pulsadores y dos LED, ¿cómo hacemos para que un pulsador encienda un LED y el otro pulsador encienda los dos LED a la vez?

Es esperable que estos desafíos desencadenen preguntas y soluciones vinculadas a la construcción de circuitos en serie y paralelo. Veamos por ejemplo una posible solución al primer problema:



Luego del pulsador, el circuito se divide en dos tramos dispuestos en paralelo, uno para cada LED.
El segundo problema puede encontrar una solución similar, aunque esta vez debamos usar el diodo que observamos al principio:

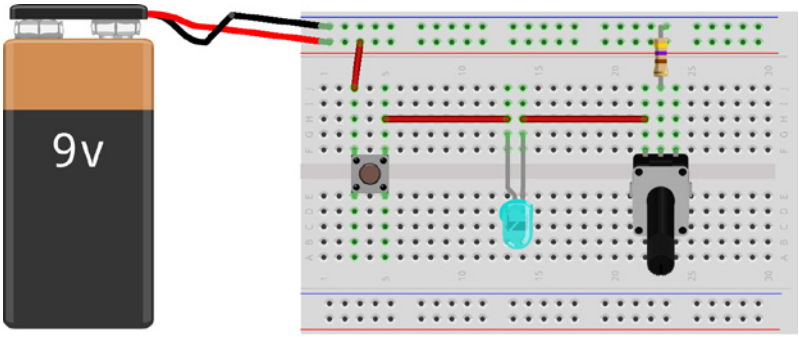


El botón de la izquierda enciende los dos LED, mientras que el botón de la derecha solo enciende el LED verde.

Por último, podemos dar un paso más en nuestra exploración de los circuitos agregando un nuevo componente electrónico, que será de utilidad en muchos proyectos: el potenciómetro.
El potenciómetro (o resistencia variable) es un componente que contiene dos resistencias cuyo valor varía cuando giramos una perilla. Su funcionamiento es muy sencillo: cuando llevamos la perilla hacia un extremo, el **cursor** (la terminal del medio) presenta una resistencia nula respecto a una de las otras terminales (A); y una resistencia máxima (determinada por el valor del potenciómetro) respecto a la otra (B). A medida que vamos girando la perilla, la resistencia cursor-A va aumentando, mientras disminuye en la misma proporción la resistencia cursor-B.

¿Cómo usamos el potenciómetro para variar la resistencia aplicada a un circuito que enciende un LED?

¡Atención!Si sustituimos por un potenciómetro la resistencia que le habíamos puesto al LED en los circuitos anteriores, podríamos destruir el LED al bajar el valor de la resistencia por debajo de los 470 Ω. Así que una solución adecuada sería colocar el potenciómetro (de 1kΩ) en serie con una resistencia fija de 470Ω, de modo que sus valores se sumen. Así, la resistencia aplicada al LED variará entre 470 y 1470Ω.



Este circuito enciende el LED azul al presionar el pulsador, con una potencia determinada por el ángulo de giro del potenciómetro.

Nombre actividad	Desarmado de basura electrónica y reconocimiento de los componentes.
Objetivo	Reconocer y recolectar componentes electrónicos para proyectos de robótica.
Áreas de conocimiento	Electrónica. Mecánica.
Conceptos relacionados	Componentes electrónicos, placas impresas, circuitos integrados, datasheet (hoja técnica).
Competencias	Resolución de problemas. Trabajo en equipo. Investigación y experimentación.
Materiales	Destornilladores (de distintas formas y tamaños), pinzas y alicates, gafas de protección ocular, guantes de trabajo, soldador a estaño, bomba para desoldar, aparatos electrónicos para destruir, computadora con internet.
Enlaces de interés	
Anexo	

Una actividad que resulta muy estimulante al momento de comenzar la construcción de proyectos de robótica es el desmantelado de basura electrónica. Este tiene el objetivo de obtener componentes que puedan resultar de utilidad, mientras aprendemos a reconocerlos y nos familiarizamos con sus aplicaciones.

El educador propondrá a los participantes que intuyan en base a sus conocimientos, mediante el ejercicio de observarlos por dentro cómo funcionan los aparatos que se están desmantelando.

Si se cuenta con las herramientas adecuadas es posible desmantelar las placas impresas para obtener unidades simples como resistencias, interruptores, transistores, potenciómetros, capacitores, diodos, motores, solenoides, circuitos integrados y muchos componentes más.

Una propuesta para comenzar con el reconocimiento es pedirles a los participantes que clasifiquen los diferentes componentes que se van obteniendo. Para hacerlo, los estudiantes se apoyarán en sus conocimientos previos o en su propia intuición, discutiendo los criterios que pueden ser, por ejemplo, de forma, de materiales, de colores, de cantidad de terminales, entre otros.

Pero, ¿cómo saber qué es cada cosa? Tal vez los participantes ya tengan idea de lo que son algunos de los componentes y puedan ir etiquetando los grupos haciendo búsquedas en internet. Para facilitar este trabajo, el educador podría traer una lista de nombres y pedirles a los participantes que los asocien con los componentes que tienen sobre la mesa, pudiendo hacer búsquedas de imágenes en internet, por ejemplo. Es posible que en este proceso sea necesario reacomodar los grupos. Tal vez haya que separar componentes que se veían similares pero eran cosas diferentes, o juntar grupos con objetos distintos a simple vista pero que llevan el mismo nombre.

Con los circuitos integrados (comúnmente llamados chips) el problema de su identificación es más sencillo e interesante a la vez, ya que suelen tener el nombre impreso en la cara superior. Eso sí, son nombres crípticos como N555 o L293D. Lo bueno es que podemos buscar esos nombres en internet y saber exactamente qué son y cómo se usan; especialmente si buscamos los famosos datasheet que son las hojas técnicas que contienen toda la información necesaria para su uso en proyectos de ingeniería.

5.3.2. Arduino

Ventajas de trabajar la Computación Física con Arduino y otras plataformas de prototipado de hardware de código abierto.

Al igual que *Beaglebone*, *Raspberry Pi*, *XBee* y otras, *Arduino* es una tecnología de hardware y software de código abierto. Esto significa que los usuarios pueden acceder a conocer todos los códigos de software y todos los parámetros relevantes en el montaje del hardware, de manera que es posible recrearlo uno mismo, agregando módulos o modificando componentes y funcionalidades. Este tipo de plataforma de hardware y software abierto ofrece muchas ventajas para el trabajo en proyectos educativos.

De la misma manera que el software de código abierto, el hardware open *source* se distribuye bajo licencias que garantizan la libertad para acceder al código fuente y modificarlo, e incluso redistribuirlo. Esto además de derivar en otras ventajas como las que veremos a continuación, tiene en sí mismo un valor relevante para los entornos educativos. Sustentadas en la colaboración de una comunidad global, las aplicaciones de software de código abierto son accesibles por provenir de diversas fuentes y fundamentalmente por no presentar límites a las posibilidades de los más curiosos para adentrarse en las diferentes soluciones que hay detrás.

Un resultado del modelo de código abierto aplicado al hardware es el bajo coste de los dispositivos de estas características que pueden ser fabricados y distribuidos por muchas firmas, sin pagar regalías por el diseño. A su vez es susceptible de modificarse para, por ejemplo, adaptarse a usos muy específicos. Además se pueden diseñar y construir nuevos dispositivos que trabajen en forma acoplada, ya que conocemos todas las especificaciones de los componentes físicos y los protocolos de comunicación aplicados.

Todas las características de este tipo de plataformas nos facilitan el poder crear nuestros propios inventos y nos invitan a experimentar sin miedo a cometer errores. Si bien los errores pueden no ser inocuos cuando trabajamos con electrónica real (si hacemos un cortocircuito o conectamos fuentes de poder a entradas equivocadas podemos quemar la placa), la experimentación en este tipo de plataformas es muy recomendable.

Con el fin de preservar el material, podemos solicitar a los participantes que eviten conectar los circuitos a la fuente de poder (puerto USB de la PC, batería o transformador) sin previa supervisión del educador. De todas formas, es bueno tener en cuenta que estaremos trabajando con una potencia eléctrica que no requiere de mayores medidas de seguridad siendo conscientes de mantener distancias adecuadas en el contacto con las fuentes de energía.

El lenguaje de programación Arduino.
El lenguaje de programación Arduino, una implementación del lenguaje C, tiene la ventaja de poder trabajar a muy bajo nivel de abstracción[13], lo que permite acercarse al lenguaje máquina y al funcionamiento de la electrónica digital, que es la base

física de toda ingeniería computacional. Por otro lado es capaz de operar al más alto nivel, desatando una gran cantidad de funciones de altísima complejidad con unas pocas líneas de código. Esto es gracias a la enorme disponibilidad de funciones embebidas y librerías [14], tanto oficiales como provistas por la comunidad. Además, este lenguaje y todas las librerías se encuentran inmensamente documentados, ejemplificados, experimentados y compartidos en internet.
Una consecuencia de lo anterior es la disponibilidad de librerías que permiten programar arduino desde diferentes lenguajes alternativos, como S4A *(Scratch)*, *Pyduino (Python) o Firmata (Java Script).*

Actividades con Arduino.

Nombre actividad	¡Hola mundo Arduino!: el led intermitente.
Objetivo	Introducirse al uso de la IDE y el lenguaje de programación de arduino.
Áreas de conocimiento	Ciencias de la computación. Electrónica.
Conceptos relacionados	Entorno Integrado de Desarrollo Arduino. Electrónica digital. Funciones en programación. Variables y tipos de datos.
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Investigación y experimentación.
Materiales	Para cada subgrupo: placa Arduino, cable USB, computadora con IDE, Arduino instalado, protoboard, leds, resistencias 220Ω, cables p/protoboard macho-macho, batería 9v con broche conector dc para arduino.
Enlaces de interés	https://www.arduino.cc/en/Main/Software https://www.arduino.cc/en/Guide/HomePage https://www.arduino.cc/en/Reference/HomePage
Anexo	

[13] Hablamos de bajo y alto nivel en la programación de software para denotar el grado de abstracción en que se encuentra determinado lenguaje o código. Dicho de otra manera, el nivel de abstracción refiere a cuán cerca o lejos estamos del lenguaje máquina (bajo nivel) o del lenguaje humano (alto nivel).
[14] Las librerías son bloques de código con una funcionalidad específica que pueden ser reutilizados en un nuevo programa para facilitar su desarrollo.
[15] Entorno integrado de desarrollo IDE por su sigla en inglés.
[16] Para descargar la IDE de Arduino ver: https://www.arduino.cc/en/Main/Software. Para una guía de instalación ver: https://www.arduino.cc/en/Guide/HomePage.

El objetivo de esta actividad es introducirnos junto a los participantes al uso de la IDE de Arduino, a la vez que vamos conociendo el lenguaje que utiliza esta plataforma para programar el microprocesador. Al finalizar habremos construido un circuito con leds que se encienden y apagan reiteradamente según un programa que vamos a escribir y cargar en la placa, con la ayuda de una computadora.

Luego de abrir la IDE de Arduino los participantes explorarán la ventana para identificar los elementos con los que vamos a trabajar:

• **El editor de texto**: Es el rectángulo blanco que ocupa casi toda la ventana. Allí escribimos y editamos el código de nuestros programas.

• **La barra de menú**, donde haremos algunas configuraciones:

- Seleccionar la placa: En el menú: Herramientas>>Tarjeta>>, seleccionamos el modelo de placa Arduino que estamos utilizando.
- Seleccionar el puerto: Con la placa conectada a la PC, vamos al menú: Herramientas>>Puerto Serial>> y seleccionamos el puerto asignado a la placa.

• **La barra de estado**, en la parte inferior de la ventana: nos indica en qué línea del código estamos posicionados(a la izquierda) y qué configuración tenemos para placa y puerto (a la derecha).

• Los botones de **la barra de herramientas** y sus funciones:

- **verificar**: Compila[17] el código para verificar si no contiene errores.
- **cargar**: Compila el código y lo carga en la placa que tengamos conectada.
- **nuevo, abrir y guardar:** Para crear un nuevo sketch[18], abrir uno que tengamos guardado, o guardar el que tenemos en la ventana.
- **monitor serial:** Es una herramienta que no utilizaremos en esta actividad pero sí más adelante. Sirve para monitorear la comunicación de la placa Arduino con la computadora.

• **El cuadro de mensajes:** Es el espacio con fondo negro que se encuentra debajo del editor de texto. Allí la computadora nos devolverá información sobre la compilación, la carga a la placa y los errores que surjan en estos procesos.

Para comenzar a experimentar cómo usamos el IDE para programar las placas Arduino abriremos el sketch de ejemplo Blink. El IDE de Arduino provee una extensa lista de sketches de ejemplo que permiten experimentar todas las funciones embebidas y librerías disponibles. Se puede acceder al sketch Blink en: Archivo>>Ejemplos>>01. Basics>>Blink, y contiene un código similar al que sigue[19]:

[17] Ciertos lenguajes de programación (como es el caso del lenguaje Arduino y sus antecesores) requieren que el código sea compilado antes de su ejecución; un proceso por el cual la máquina construye en base a las instrucciones escritas por el humano, un archivo binario (compuesto únicamente de ceros y unos) que puede ser ejecutado directamente por el microprocesador.

[18] Un sketch (retazo o fragmento) es una sucesión de líneas de código que pueden ser guardadas en un documento de texto plano y ejecutarse como una unidad, o formar parte de un programa mayor que integre sus variables y funciones.

[19] Algunos parámetros o comentarios pueden variar según la versión de IDE que estemos usando.

```
/*
  Blink
  Enciende un led por un segundo, luego lo apaga por un segundo, repetidamente.
  Este código de ejemplo es de dominio público.
*/

// El pin 13 tiene un led conectado en la mayoría de las placas arduino.
// démosle un nombre:
int led = 13;

// la secuencia setup se ejecuta una sola vez cuando presionas reset:
void setup() {
        // inicializa el pin digital "led" como una salida.
        pinMode(led, OUTPUT);
}

// la secuencia loop se ejecuta una y otra vez por siempre:
void loop() {
        digitalWrite(led, HIGH);  // enciende el LED (HIGH significa nivel de voltaje alto)
        delay(1000);              // espera durante un segundo
        digitalWrite(led, LOW)    // apaga el LED (LOW significa nivel de voltaje bajo)
        delay(1000);              // espera durante un segundo
}
```

Los participantes notarán que el texto está señalado con diferentes colores. La idea es observar entre todos los siguientes elementos para qué sirven y cómo se escriben:

• **Comentarios:** Todo el texto señalado con el color gris[20] sirve para agregar comentarios para las diferentes partes del código con el objetivo de que los usuarios podamos comprender o recordar qué hace cada línea o sección del sketch. La máquina reconoce estos comentarios por encontrarse luego de dos barras (//) en una línea, o entre /* (abrir comentario) y */ (cerrar comentario) en múltiples líneas, y simplemente los pasa por alto durante la compilación. Nótese que es posible modificarlos e incluso eliminarlos sin que nada cambie en el comportamiento del programa.

[20] Los colores pueden variar según la versión del IDE o la configuración del usuario.

• **Declaración de variables** con **tipo de dato entero** (int): la instrucción **int** crea una variable que contiene un número entero, que en este caso toma el nombre "led" y el valor 13[21]. Adivinanza: ¿Por qué le ponemos nombre al pin si podemos usar el número que ya tiene?[22]

• **Declaración de funciones void:** Void es un tipo de dato vacío. Se usa para declarar funciones de las que no se espera que devuelvan algún valor.

• **setup**() y **loop**(), las dos funciones imprescindibles de Arduino: estas funciones marcadas en verde (setup y loop) son funciones principales que debemos declarar necesariamente en cualquier programa de Arduino.

- La función void setup() ejecuta las instrucciones que pongamos entre las llaves { } UNA SOLA VEZ cuando conectamos la energía a la placa.
- La función void loop() ejecuta las instrucciones que pongamos entre las llaves { } UNA Y OTRA VEZ hasta que desenchufemos la placa o la reiniciemos presionando el botón reset.

• **Funciones embebidas, pinMode(), digitalWrite(), delay():**

- Al interior de setup() encontramos la función **pinMode**(), que formatea los puertos (pin) de Arduino como de entrada de información (INPUT) o salida de información (OUTPUT). Esta función requiere que le pasemos dos parámetros (entre los paréntesis): primero a qué pin nos referimos (en este caso al pin 13, ya que al principio declaramos que *led* = 13) y luego de la coma, en qué modo lo queremos configurar, si como entrada o salida (en este caso salida).
- Al interior de loop() encontramos la función **digitalWrite**() que hace una escritura digital. Produce una salida de información binaria (corriente eléctrica alta o baja) hacia un pin (en este caso "led") y los dispositivos de salida que le conectemos. El primer parámetro que espera esta función es el pin al que nos referimos. El segundo es qué nivel de corriente queremos que escriba (alto o bajo; HIGH o LOW).
- También en loop() encontramos la función **delay**(), que detiene la ejecución de la secuencia por el tiempo que le pasemos como parámetro, en milisegundos (en este caso 1000; un segundo).

[21] Las variables en el lenguaje de programación Arduino tienen un tipo único de dato asociado. int crea un entero, float crea un decimal, char un caracter, string una cadena de caracteres... Para una lista completa de los tipos de dato en Arduino ver Data Types en: http://www.arduino.org/learning/reference.

[22] Una respuesta: Es buena práctica usar nombres mnemotécnicos para nuestros objetos. Es decir llamemos a las cosas con nombres que nos recuerden lo que son. Entonces si tenemos un LED, ¡llamémosle "led"!
Pero hay otra razón: ¿qué tal si necesitamos cambiar el pin al que tenemos conectado el LED? Ciertamente es mejor cambiar el valor en una expresión al principio, que recorrer todo el programa buscando y sustituyendo cada ocasión en que el número 13 refiere al pin. ¡Un desastre si el programa es algo complejo!

• **Parámetros especiales:** El texto que aparece en color azul consiste en palabras reservadas por el lenguaje para la declaración de variables o funciones y para la inserción de parámetros en algunas funciones, como HIGH o LOW en digitalWrite() o OUTPUT e INPUT en pinMode().

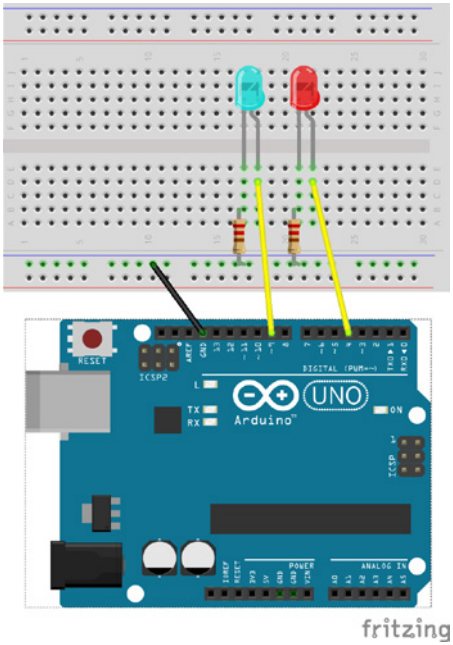
A medida que vamos repasando el código, debemos animar a los participantes a experimentar modificando cosas y alterando los valores; ello puede ser una manera de ir descubriendo qué hace cada función:

• ¿Qué sucede si cambiamos el número dentro de la función delay()?

• ¿Cómo podemos hacer para que el led se encienda durante un segundo, se apague durante un segundo y luego se encienda durante una décima de segundo y se apague durante una décima de segundo, antes de volver a comenzar el loop?

Para completar una visión acerca de cómo construimos prototipos con Arduino recomendamos experimentar con otros led, además del que viene incluido en la placa, agregando circuitos con ayuda de la protoboard.

Si se cuenta con conexión a internet, es buena idea proponer a los participantes que busquen ellos cómo se conecta un led a Arduino. Es realmente fácil encontrar imágenes, tutoriales y videos de gente que comparte sus proyectos y sus conocimientos.
Aquí un ejemplo de circuito con un par de leds conectados a las clavijas digitales 4 y 9:



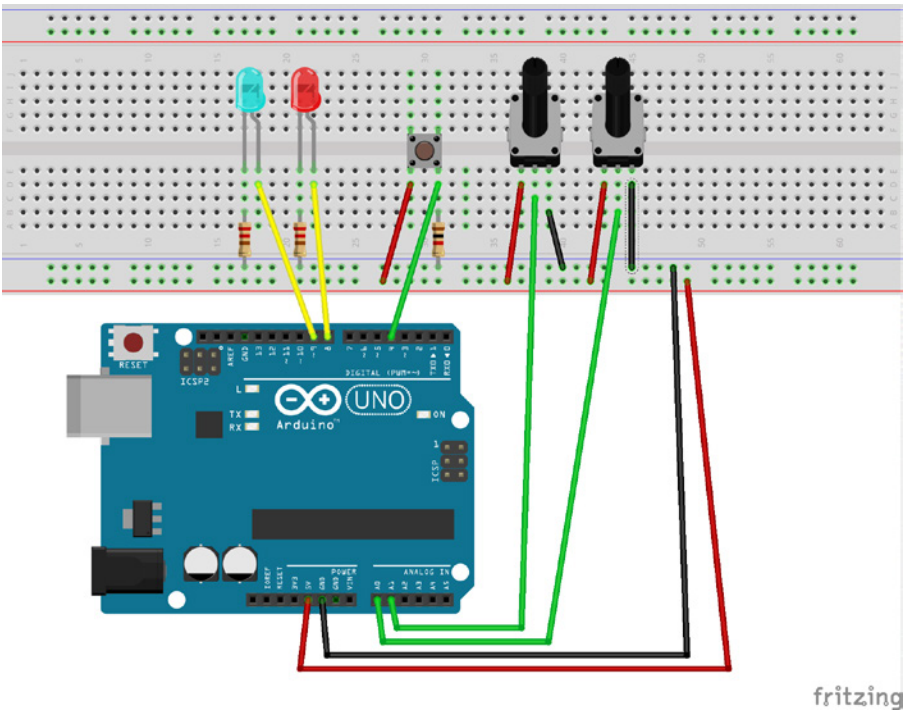
¡Programemos una divertida secuencia utilizando estos dos leds y algunos más!

Nombre actividad	Primeros sensores: interruptor y potenciómetro.
Objetivo	Introducirnos al lenguaje arduino. Experimentar con funciones de lectura digital y analógica. Construir un dispositivo con sensores y actuadores.
Áreas de conocimiento	Ciencias de la computación. Electrónica
Conceptos relacionados	Lectura digital. Lectura analógica. Dispositivos de entrada y salida (sensores y actuadores). Resistencia eléctrica - Ley de Ohm. Circuitos eléctricos. Lenguaje de programación arduino
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Investigación y experimentación
Materiales	Para cada subgrupo: 1. Computadora con IDE instalado 2. Placa arduino 3.Cable USB 4. Protoboard 5. Cables macho-macho para protoboard 6. 2 potenciómetros de 1-10KΩ 7. Interruptor 8. 2 leds 9. Resistencia 10kΩ 10. 2 resistencias 220Ω 11. Batería 9v con broche conector dc para arduino
Enlaces de interés	https://www.arduino.cc/en/Reference/HomePage
Anexo	

Esta actividad puede plantearse como una continuación de la anterior, donde trabajando con diodos emisores de luz (LED) construimos un dispositivo que actúa sobre el mundo físico produciendo luces intermitentes. Al finalizar habremos construido un dispositivo cuyos actuadores (los LED) realizan sus acciones (encender y apagar) en función de la información que devuelven los sensores: un interruptor y dos potenciómetros.

- El interruptor es un sensor digital, ya que solo puede devolver dos estados: encendido (voltaje alto) y apagado (voltaje bajo).
- Al potenciómetro en cambio lo usaremos como sensor analógico, ya que nos devolverá un dato que puede ubicarse en cualquier punto entre los niveles máximos y mínimos de voltaje.

El circuito que debemos montar es un tanto más complejo que el anterior, por lo que el educador puede optar por proponer a los participantes ir construyéndolo por partes, alternando la programación de las funciones para cada componente.



Comencemos con los LED. Si el grupo ya ha realizado la actividad anterior, podemos proponerles que reconstruyan el circuito y recuperen el programa que habíamos editado. También usamos una variable de tipo de dato entero para darle un nombre al pin donde conectamos los LED. Las variables que creamos en la actividad anterior sirvieron como constantes, ya que guardamos en ellas valores que no debían variar durante la ejecución. Podemos explicitar que declaramos una constante, y no una variable, agregando la expresión “const” directamente antes de “int” o cualquier otro tipo de dato.

Luego de editar un poco el sketch de la actividad anterior quedaría similar al que sigue:

```
// le damos nombre a los leds usando constantes:
const int led_azul = 9;
const int led_rojo = 8;

//cremaos un par de variable para el tiempo de los delay:
int tiempo_azul = 1000;
int tiempo_rojo = 1000;

void setup() {
// inicializamos los pines de los leds como salidas.
 pinMode(led_azul, OUTPUT);
 pinMode(led_rojo, OUTPUT);
}

void loop() {
 digitalWrite(led_azul, HIGH);  //enciende el led azul...
 digitalWrite(led_rojo, LOW);   //...y apaga el rojo.
 delay(tiempo_azul);            //espera durante un tiempo
 digitalWrite(led_azul, LOW);   //apaga el led azul...
 digitalWrite(led_rojo, HIGH);  //...y enciende el rojo.
 delay(tiempo_rojo);            //espera durante un tiempo
}
```

Si alteramos los valores de las variables "tiempo_azul" y "tiempo_rojo" antes de volver a cargar el programa en la placa obtendremos secuencias distintas.
Ahora daremos un salto en el uso de las variables que hará evidente lo importantes que son en la programación. Haremos que el valor que guardan se actualice cada vez que se repite el loop, y le diremos a los LED que actúen en función de esos cambios. ¿Qué tal si introducimos el primer sensor?

El sensor digital.

Experimentemos con un sensor digital instalando un interruptor en nuestro circuito. Como se ve en la imagen, el interruptor recibe en uno de sus extremos una corriente (5v). Cuando presionamos sobre el interruptor esta corriente pasa al pin digital 4 y a la tierra a través de una resistencia (10kΩ). La función que cumple esta resistencia en el circuito se llama "pull-down" y es de vital importancia. Si no tuviésemos esa descarga a tierra, el Conversor Analógico Digital[23] de Arduino no tendría una referencia para saber si el voltaje que recibe es alto o bajo, dando lugar a lecturas erróneas. Por otro lado si en vez de usar una resistencia conectásemos el interruptor a tierra mediante un cable, estaríamos creando un cortocircuito al presionar el botón.

[23] El ADC (analog digital converter) y el DAC (digital analog converter) son circuitos integrados, presentes en las placas Arduino, que permiten transformar señales analógicas (como los voltajes que provienen del mundo real) a digitales y viceversa.

Luego de instalado el interruptor incluyámoslo en nuestro *sketch:*

Agregamos a las constantes declaradas una para el pin donde conectamos el interruptor:

```
const int boton = 4;
```

En la zona de las variables agregamos una donde guardaremos la lectura que hagamos del estado del interruptor. Como decíamos, esta variable solo podrá asumir dos valores, así que usaremos otro tipo de dato pensado especialmente para estos casos; el tipo de dato booleano:

```
boolean encender = 0;
```

Ajustamos el pin del interruptor como una entrada:

```
pinMode(boton, INPUT);
```

Haremos que nuestro programa lea constantemente el estado del interruptor para decidir si ejecuta la secuencia o apaga los dos leds. Tendremos que construir otro código para el loop que contendrá la secuencia que ya tenemos escrita pero agregará algunas cosas más. ¿Qué tal si declaramos una nueva función que contenga lo que hasta ahora es nuestro loop y la invocamos en el nuevo loop que escribamos?
Lo podemos hacer de una manera muy sencilla: sustituimos la palabra "loop" por otra no reservada por el lenguaje, por ejemplo "secuencia". De paso podemos escribir una función para la otra eventualidad, cuando no estamos presionando el botón:

```
void secuencia(){
  digitalWrite(led_azul, HIGH);  //enciende el led azul...
  digitalWrite(led_rojo, LOW);   //...y apaga el rojo.
  delay(tiempo_azul);            //espera durante un tiempo
  digitalWrite(led_azul, LOW);   //apaga el led azul...
  digitalWrite(led_rojo, HIGH);  //...y enciende el rojo.
  delay(tiempo_rojo);            //espera durante un tiempo
}
void apagar_todo(){
  digitalWrite(led_azul, LOW);  //apaga el led azul
  digitalWrite(led_rojo, LOW);  //apaga el led rojo
}
```

Estas nuevas funciones que hemos declarado quedarán al final del sketch. Antes, luego del setup() reescribiremos nuestro loop(). Usaremos aquí una estructura de control: "if.. else.." (si... sino...). El loop quedará similar al que sigue:

```
void loop(){
  encender = digitalRead(boton);  //leemos el estado del interruptor y lo guardamos en la
                                  //variable "encender" que creamos para eso
  if(encender == 1){      //si es verdadero que encender = 1...
    secuencia();          //ejecuta la función secuencia().
  }
  else{                   //De lo contrario...
    apagar_todo();        //ejecuta la función apagar_todo()
  }
}
```

En el parámetro de la función if() hemos puesto una condición, que es lo que esta función requiere para saber si debe ejecutar, o no, lo que está dentro de las llaves {}. Nótese que al expresar esa condición usamos dos signos de igual juntos. Es que para los lenguajes de programación basados en C, como en otros, un solo signo de igual significa atribuirle el valor a la variable, no compararlos. ¡Poner un solo signo cuando queremos comparar es un error muy común al que debemos estar atentos!

El sensor analógico.

Es hora de completar nuestro circuito. Vamos a conectar dos potenciómetros que pueden ser de 1 a 10KΩ. Como los potenciómetros son resistencias variables, sus medidas se representan con la misma unidad que las resistencias normales (los Ohmios), pero en el caso de los potenciómetros la medida expresa su resistencia máxima.

Conectar un potenciómetro a Arduino para que funcione como sensor analógico es muy fácil. La terminal que se encuentra en el centro (el cursor) es la que debemos conectar a alguno de los pines de entrada analógica de la placa Arduino. Las otras dos terminales, las de los extremos, se llaman control de tensión y debemos conectarlas una a la corriente (5v) y la otra a la tierra. Cuál de las dos va a la tierra no importa mucho; solo cambiará el sentido en que debemos girar la perilla para que la resistencia aumente o disminuya.

Podemos rápidamente probar las lecturas analógicas mediante el sketch "AnalogReadSerial" que encontramos en los sketches de ejemplo del IDE de Arduino (Archivo>>Ejemplos>>01.Basics>>AnalogReadSerial). Es además una oportunidad para experimentar con el Monitor Serial. Este sketch de ejemplo mide una entrada analógica conectada al pin A0 e imprime el dato en el puerto serial. Si lo cargamos a la placa y luego (sin desconectarla de la computadora) abrimos el monitor serial

veremos una ventana con unos números que van cayendo en cascada. ¡Cada número que se agrega es una nueva lectura del sensor que el Arduino ha enviado a la computadora mediante la conexión serial!

Instalados los potenciómetros en el circuito y probados mediante el sketch de ejemplo, es hora de incluirlos en nuestro programa. Agregamos una constante para cada conexión:

```
const int pote1 = 0;
const int pote2 = 1;
```

Las variables para guardar los valores de lectura de los potenciómetros:

```
int sensor_pote1;
int sensor_pote2;
```

No es necesario que usemos la función pinMode() para configurar las conexiones como entradas, ya que los pines analógicos de Arduino no pueden funcionar como salidas[24].
¡Tendremos unas perillas que controlan el ritmo! Probemos agregar a nuestro loop las funciones de lectura analógica que tomen el dato de los potenciómetros para determinar aquellas variables que declaramos para el tiempo de luz de los LED:

```
void loop(){
  encender = digitalRead(boton);  //Leemos el estado del interruptor y lo guardamos en la
                                  //variable "encender" que creamos para eso.
  sensor_pote1 = analogRead(pote1);//Leemos estado de potenciometros y guardamos
  sensor_pote2 = analogRead(pote2); //los datos en las variables que creamos para eso.
  tiempo_azul = sensor_pote1;  //igualamos la variable tiempo_azul al sensor 1.
  tiempo_rojo = sensor_pote2;  //la variable tiempo_rojo al valor de lectura del sensor 2.
  if(encender == 1){      //si es verdadero que encender = 1...
    secuencia();          //ejecuta la función secuencia().
  }
  else{                   //De lo contrario...
    apagar_todo();        //ejecuta la función apagar_todo()
  }
}
```

[24] La función de escritura analógica utiliza los pines digitales como veremos más adelante.

Cada vez que se ejecute la función secuencia (), lo hará con los valores que las variables tiempo rojo y tiempo azul han tomado de las lecturas de los sensores al comenzar el loop. ¡Genial! Pero, ¿cuánto tiempo es eso en realidad? Así las cosas, el tiempo de encendido de cada LED tomará el valor de lectura del potenciómetro. Esta última varía entre 0 y 1023, así que el tiempo oscilará entre 0 y 1023 milisegundos. ¿Y qué si quiero hacer que duren algo más que 1,023 segundos?
Una buena idea es utilizar una hermosa función de Arduino: la función map(), cuya sintaxis puede expresarse de la siguiente manera:

```
resultado = map(entrada, min_rango_origen, max_rango_origen, min_rango_destino, max_rango_destino);
```

Si entendemos lo que hace no resulta tan complicado. El valor que asumirá la variable resultado equivale al lugar que ocupa la variable entrada en el rango_origen, pero trasladado al rango_destino.

Apliquemos esta potente función para hacer que el tiempo de encendido de los LED varíe entre 0,1 y 5 segundos. Al final tendremos un programa como este:

```
const int led_azul = 9;
const int led_rojo = 8;
const int boton = 4;
const int pote1 = 0;
const int pote2 = 1;

int tiempo_azul;
int tiempo_rojo;
boolean encender;
int sensor_pote1;
int sensor_pote2;

void setup() {
  pinMode(led_azul, OUTPUT);
  pinMode(led_rojo, OUTPUT);
  pinMode(boton, INPUT);
}

void loop(){
   encender = digitalRead(boton);
   sensor_pote1 = analogRead(pote1);
   sensor_pote2 = analogRead(pote2);
   tiempo_azul = map(sensor_pote1, 0, 1023, 100, 5000);
   tiempo_rojo = map(sensor_pote2, 0, 1023, 100, 5000);
   if(encender == 1){
      secuencia();
   }
```

```
  else{
    apagar_todo();
  }
}

void secuencia(){
  digitalWrite(led_azul, HIGH);
  digitalWrite(led_rojo, LOW);
  delay(tiempo_azul);
  digitalWrite(led_azul, LOW);
  digitalWrite(led_rojo, HIGH);
  delay(tiempo_rojo);
}

void apagar_todo(){
  digitalWrite(led_azul, LOW);
  digitalWrite(led_rojo, LOW);
}
```

Nombre actividad	¡A mover! Motores, transistor y puente H.
Objetivo	Introducirnos al lenguaje arduino. Experimentar con el uso de circuitos integrados. Construir un dispositivo actuador que recibe comandos mediante comunicación serial.
Áreas de conocimiento	Ciencias de la computación. Electrónica
Conceptos relacionados	Escritura digital y analógica. Modulación por ancho de pulsos (PWM). Transistor. Capacitor eléctrico. Circuitos integrados
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Investigación y experimentación
Materiales	Para cada subgrupo: Computadora con IDE instalado Placa Arduino Cable USB Protoboard Cables macho-macho para protoboard Puente H: L293D o SN754410 2 mini motores DC 9V 1 transistor MOSFET 1 resistencia 220Ω Batería 9v con broche conector para protoboard
Enlaces de interés	https://www.arduino.cc/en/Reference/HomePage
Anexo	

Hasta ahora hemos construido dispositivos con Arduino cuya actuación consiste en encender algunos LED. Algo emocionante sin dudas pero un poco limitado si pensamos en proyectos de robótica. Lo bueno es que con lo que hemos aprendido ya podemos programar algunos actuadores muy potentes, como un motor que abre un portón, por ejemplo. Solo es cuestión de construir los circuitos adecuados, y del mismo modo que encendemos un LED, podremos hacer algunas cosas ciertamente más interesantes.

La mayoría de los pequeños motorcitos que encontramos en muchos juguetes electrónicos pueden funcionar a 5v. Podríamos conectarlos directamente a las clavijas de la placa Arduino como hicimos con los LED. ¡Mejor no! Es que la corriente continua no solo se hace de una *diferencia de potencial* (que se mide en Voltios) sino que además se expresa en una *intensidad* (que se mide en Amperios). Las clavijas digitales de las placas Arduino pueden proveernos de unos 200mA, que son suficientes para encender varios LED pero que no le harían cosquillas incluso a un motor eléctrico pequeño, y nos quedaríamos sin corriente para alimentar el microprocesador y los circuitos de la propia placa.

Tenemos que alimentar a nuestros motores (o cualquier otro actuador que requiera cierta potencia) con una fuente de poder diferente a la que alimenta la placa Arduino, pero usar la señal débil de los pines digitales para controlarlos. Un problema muy básico que podemos solucionar fácilmente gracias al que tal vez sea el invento electrónico más impactante del siglo XX: **el transistor.**

El transistor es un componente electrónico que permite controlar una corriente eléctrica. Gracias a este importante dispositivo ha sido posible construir computadoras potentes de tamaño manejable. Antes de ser inventado era necesario usar válvulas de vacío con un sinnúmero de inconvenientes que incluían el alto consumo de energía, el calor extremo, el gran tamaño, el alto costo y la baja durabilidad.

Existen diversos tipos, pero los que usaremos cuentan con tres terminales fundamentales:

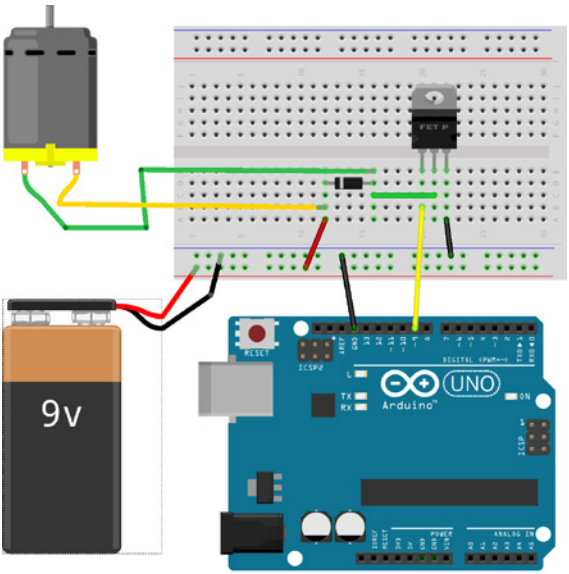
- una **compuerta**: que es donde conectamos la señal con la que vamos a controlar el flujo.
- un **drenaje**: que es la corriente que sale en función de la señal que llega a la compuerta.
- una **fuente**: que es la entrada de la corriente que queremos controlar.

Para alimentar un motor con una batería de 9V y controlarlo con Arduino podemos usar un transistor MOSFET. En este ejemplo conectaremos una terminal del motor al polo positivo de la batería y pondremos el transistor del lado de la tierra. Cuando apliquemos corriente a la compuerta el transistor conectará la fuente con el drenaje produciendo descarga a tierra y encendiendo el motor.

Además conviene usar un **diodo** para evitar el retorno de energía generada por el motor hacia los circuitos y la placa Arduino. Es que el motor no solo puede transformar corriente eléctrica en trabajo mecánico; también a la inversa, puede producir corriente eléctrica si aplicamos fuerza mecánica al eje, al igual que la turbina de un generador. Cuando cortamos la corriente de un motor que está encendido, es posible que este siga moviéndose por un tiempo debido a la inercia y envíe por

los circuitos una corriente contraria a la que estamos aplicando. Como vimos con los LED, los diodos son componentes polarizados; en un sentido son conductores y en el otro son no-conductores. La línea blanca a un lado del diodo indica su lado negativo, hacia donde la corriente puede ir, pero de donde no puede volver.

La utilidad del diodo en este caso es que una posible corriente inversa pase directamente desde una terminal a la otra del motor sin generar una diferencia de potencial:



Podemos ponerlo a prueba con un pequeño código, muy parecido a nuestro conocido *sketch blink:*

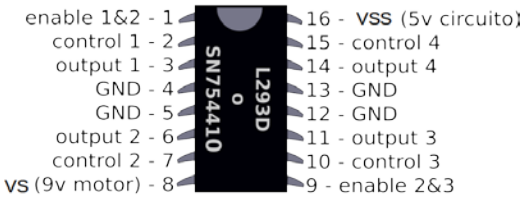
```
const int motor = 9;

void setup() {
pinMode(motor, OUTPUT);
}

void loop() {
analogWrite(motor, 255);
delay(1000);
analogWrite(motor, 190);
delay(1000);
analogWrite(motor, 100);
delay(1000);
analogWrite(motor, 0);
delay(1000);
}
```

Hay una diferencia significativa entre este blink y el que usamos con los LED. Estamos usando la función analogWrite(), que realiza una escritura analógica. O sea que le podemos decir a un pin de Arduino que provea un voltaje intermedio en vez de solamente optar entre todo y nada como hacíamos con digitalWrite(). Debemos expresar ese nivel de energía mediante un número entero entre 0 y 255.

¡Importante!: no todos los pines digitales de Arduino están facultados para hacer escrituras analógicas. Debemos buscar los que están marcados como PWM, que significa modulación por ancho de pulsos (Pulse Width Modulation, PWM por su sigla en inglés), el método con que Arduino construye (o simula[25]) la salida analógica.

Estamos moviendo motores con programación, siendo este un paso previo a la construcción de un robot. Daremos un nuevo salto en nuestras competencias en electrónica, al usar nuestro primer circuito integrado: el puente H. Se trata de un circuito que integra varios transistores, diodos y capacitores de manera que nos permite cambiar la polaridad de la corriente que alimenta un motor, alterando su dirección de giro, como experimentamos más arriba en las actividades de electrónica analógica.

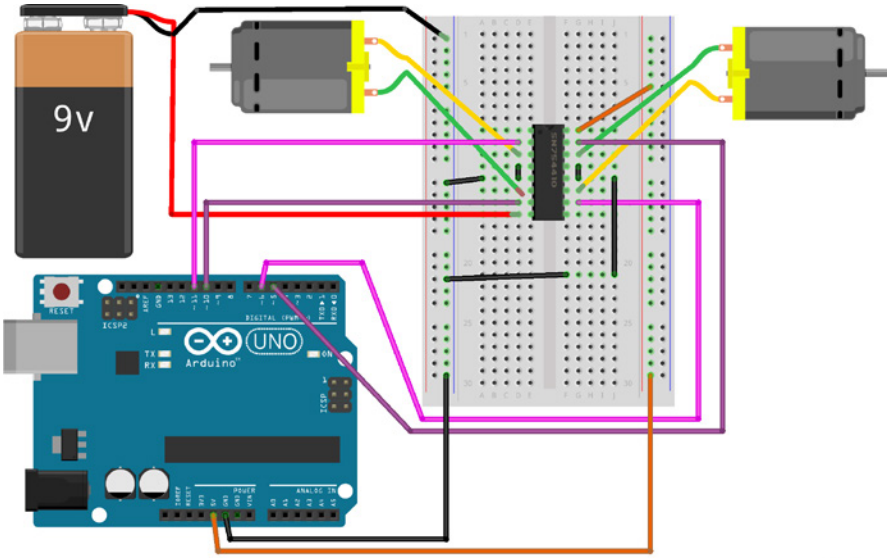
El chip que utilizaremos aquí (L293D o SN754410) es en la práctica un doble puente H, que permite controlar dos motores, uno a cada lado. El secreto con los circuitos integrados es interpretar sus datasheet, que son unos documentos que recogen toda la información relevante sobre su funcionamiento. Se pueden encontrar fácilmente en internet poniendo el código del integrado en el buscador. Aunque si solo queremos saber cómo conectarlo tal vez no sea necesario leer la hoja técnica, ya que estos integrados son muy usados y hay muchos dibujos esquemáticos que nos muestran cómo hacerlo.



[25] Es que Arduino en verdad no puede producir salidas analógicas. El microprocesador solo recibe, procesa y devuelve información digital (unos y ceros). Al igual que en el cine se simula el movimiento mediante una rápida sucesión de imágenes fijas, el Conversor Digital Analógico de Arduino enciende y apaga muy rápidamente sus salidas, produciendo pulsos. Modulando el ancho de dichos pulsos en el tiempo se produce en muchos actuadores un efecto similar al de modular el voltaje de manera analógica.

Las terminales de los circuitos L293D y SN754410 son idénticas:

- enable (pin 1 y 9): sirven para activar o desactivar las salidas al motor. Pueden dejarse sin conectar.
- output (pin 3, 6, 11, 14): es donde conectamos los motores, uno a cada lado.
- control (pin 2, 7, 10 y 15): sirven para controlar el encendido y la dirección de giro de los motores.
- GND (pin 4, 5, 12, 13): van conectados a tierra.
- VS (pin 8): es la corriente que alimenta los motores. Lo conectamos al polo positivo de la batería.
- VSS (pin 16): la corriente que alimenta el circuito integrado. Lo conectaremos a los 5v de Arduino.



En este caso estamos alimentando dos motores con una batería de 9v y los controlamos con las clavijas 5 y 6 de un lado (motor de la derecha) y 10 y 11 por otro (motor de la izquierda). Si encendemos la clavija 11 y apagamos la 10, el motor de la izquierda girará en una dirección. Si apagamos la clavija 11 y encendemos la 10, girará en la dirección contraria. Si encendemos o apagamos ambas el motor izquierdo permanecerá apagado. Lo mismo para los pines 5 y 6 y el motor derecho. Nótese que las conexiones de tierra de la placa Arduino y de la batería se encuentran juntas en la protoboard. Esto es importante cuando trabajamos con circuitos con más de una fuente de poder; los polos positivos son diversos, pero la tierra (polo negativo) debe ser única, ya que es la referencia (donde se establece el 0) para saber y comparar las diferencias de potencial (voltajes).

Muchos notarán que el circuito que acabamos de construir está muy cerca de transformarse en un verdadero robot. Si montamos este circuito sobre una interfaz física (como un rectángulo de madera y dos ruedas) y logramos controlar los motores tendremos un robot que puede moverse por el espacio y ser la base para asumir otras funcionalidades. Por ejemplo podríamos agregar sensores de distancia que permitan identificar la presencia de obstáculos y sortearlos (o atacarlos si queremos un robot SUMO). Como aún no hemos agregado ningún sensor, podemos programar nuestro robot para que haga una rutina preestablecida.

Un desafío: ¿Cómo programamos este robot para que se desplace dibujando un cuadrado?

Una posibilidad:

```
const int motorDerecho_avanzar = 5;
const int motorDerecho_retroceder = 6;
const int motorIzquierdo_avanzar = 10;
const int motorIzquierdo_retroceder = 11;

int tiempo_de_rotacion = 4000;  //¿cuánto tarda nuestro robot en dar un giro de 360°?

void setup(){
  pinMode(motorDerecho_avanzar, OUTPUT);
  pinMode(motorDerecho_retroceder, OUTPUT);
  pinMode(motorIzquierdo_avanzar, OUTPUT);
  pinMode(motorIzquierdo_retroceder, OUTPUT);
}

void loop(){
  avanzar();
  delay(2000);
  girar_derecha();
  delay(tiempo_de_rotacion / 4);
}

void avanzar(){
  digitalWrite(motorDerecho_avanzar, HIGH);
  digitalWrite(motorDerecho_retroceder, LOW);
  digitalWrite(motorIzquierdo_avanzar, HIGH);
  digitalWrite(motorIzquierdo_retroceder, LOW);
}

void girar_derecha(){
  digitalWrite(motorDerecho_avanzar, LOW);
  digitalWrite(motorDerecho_retroceder, HIGH);
  digitalWrite(motorIzquierdo_avanzar, HIGH);
  digitalWrite(motorIzquierdo_retroceder, LOW);
}
```

Nombre actividad	Comunicación Serial
Objetivo	Construir un dispositivo que responde a comandos enviados mediante comunicación serial.
Áreas de conocimiento	Electrónica. Ciencias de la computación. Telecomunicaciones
Conceptos relacionados	Comunicación entre dispositivos. Utilización de protocolos
Competencias	Resolución de problemas. Programación. Trabajo en equipo. Investigación y experimentación
Materiales	Para cada subgrupo: Computadora con IDE instalado Placa arduino Cable USB Protoboard Cables puente para protoboard Puente H: L293D o SN754410 2 mini motores DC 9V 1 módulo bluetooth (HC-05 o HC-06) 1 resistencia 5,6kΩ 1 resistencia 10kΩ Batería 9v con broche conector para protoboard
Enlaces de interés	
Anexo	

Tenemos un robot que se desplaza dibujando un cuadrado, es un robot muy hábil pero lo que hace no parece muy inteligente. ¿Qué tal si programamos algunas funciones más (parar, retroceder, girar a la izquierda) para que obedezca otras órdenes?

Podemos usar para eso la Comunicación Serial, que es un protocolo que maneja la placa Arduino para comunicarse con otros dispositivos como la computadora. Aquí haremos uso nuevamente del Monitor Serial, que experimentamos más arriba con el sketch de ejemplo "AnalogReadSerial". En aquella instancia usamos el monitor para visualizar los datos que enviaba la placa Arduino a la computadora. Ahora lo usaremos a la inversa; enviaremos comandos desde la computadora a la placa.

Veamos con atención una posible solución a este problema para identificar las funciones que usamos para la comunicación serial:
Primero agregamos al sketch de la actividad anterior las funciones de movimiento que faltaban:

```
void parar(){
  digitalWrite(motorDerecho_avanzar, LOW);
  digitalWrite(motorDerecho_retroceder, LOW);
  digitalWrite(motorIzquierdo_avanzar, LOW);
  digitalWrite(motorIzquierdo_retroceder, LOW);
}

void girar_izquierda(){
  digitalWrite(motorDerecho_avanzar, HIGH);
  digitalWrite(motorDerecho_retroceder, LOW);
  digitalWrite(motorIzquierdo_avanzar, LOW);
  digitalWrite(motorIzquierdo_retroceder, HIGH);
}

void retroceder(){
  digitalWrite(motorDerecho_avanzar, LOW);
  digitalWrite(motorDerecho_retroceder, HIGH);
  digitalWrite(motorIzquierdo_avanzar, LOW);
  digitalWrite(motorIzquierdo_retroceder, HIGH);
}
```

Luego debemos agregar una línea al interior de setup():

```
  Serial.begin(9600);
```

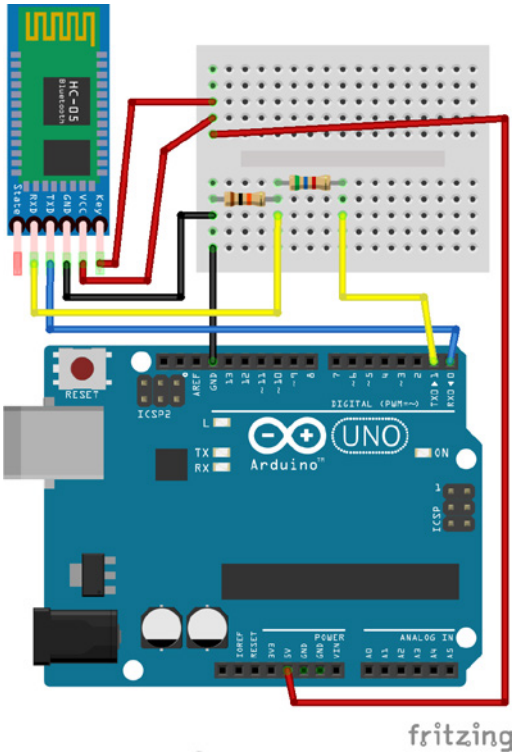
Este comando inicia la comunicación serial, estableciendo la velocidad de comunicación en 9600 baudios (bits por segundo). Este número es un standard que ya se encuentra seteado en el monitor serial. Si determinamos otra velocidad de conexión (máximo 115200) debemos tener la precaución de especificar dicha velocidad en el dispositivo que se encuentre al otro lado de la comunicación, en este caso el monitor serial en la computadora.

Ahora sí, vamos a reescribir el loop() para que escuche cualquier dato que llegue al puerto serial, los compare con los comandos que estableceremos y ejecute las funciones correspondientes:

```
void loop(){
  if(Serial.available()){          // Si hay datos disponibles en el puerto serial...
    char comando;                  // crea una variable de tipo de dato caracter...
    comando = Serial.read();  // llena la variable con el primer caracter en el puerto serial
    if(comando == 'a'){ avanzar();}
    else if(comando == 'r'){ retroceder();}
    else if(comando == 'p'){ parar();}
    else if(comando == 'd'){ girar_derecha();}
    else if(comando == 'i'){ girar_izquierda();}
    else if(comando == 10){ ;}   // no hacer nada si el caracter es un Enter
    else{ Serial.println("comando desconocido");} // devuelve un mensaje si el comando
                                                  // no es ninguno de los anteriores.
  }
}
```

Luego de construido nuestro nuevo sketch y cargado en la placa, podemos abrir el monitor serial e introducir nuestros comandos. Si en el cuadro "enviar" del monitor serial escribimos por ejemplo una letra 'd' y luego presionamos el botón enviar, nuestro robot comenzará a girar hacia la derecha, hasta que enviemos otro comando. Algo muy estimulante, pero debemos ir con la computadora detrás del robot, ya que nuestros comandos viajan a través del cable USB. ¡Eliminémoslo!

Usaremos un nuevo circuito integrado: un módulo Bluetooth, que permite generar una comunicación serial de manera inalámbrica. Una forma muy práctica (sin modificar el programa que hemos hecho) es instalar en nuestro circuito el módulo Bluetooth utilizando los pines digitales 1 y 2 de Arduino, que están reservados para la comunicación serial mediante el puerto USB[26]. Para ello debemos agregar a los circuitos de nuestro robot el módulo Bluetooth de la siguiente manera:



[26] El inconveniente que tiene esta solución es que estamos utilizando el mismo puerto para las conexiones USB y Bluetooth, por lo que generaríamos conflicto si pretendemos usarlas a la vez. No hay problema si desconectamos la corriente del módulo Bluetooth si necesitamos conectarla por USB para cargar el programa, por ejemplo. Otra solución a este problema de superposición es usar la librería "SoftwareSerial" que viene incluida en la IDE, con la que podemos crear una comunicación serial usando otros pines digitales como puerto.

La conexión es igual si en vez del módulo HC-05 tenemos un HC-06, salvo que este último no tiene las clavijas de los extremos *(State y Key)*.

Ahora podemos usar un dispositivo que tenga conectividad por Bluetooth para mandarle los comandos a nuestro robot. Si contamos con un celular o tableta con sistema operativo Android, por ejemplo, podemos instalar desde la Play Store alguna de las muchas aplicaciones gratuitas que permiten enviar caracteres mediante Bluetooth. Una buena opción es usar la app "Bluetooth Electronics" de Keuwlsoft, que permite crear paneles personalizados para enviar comandos y mostrar y graficar los datos entrantes[27].

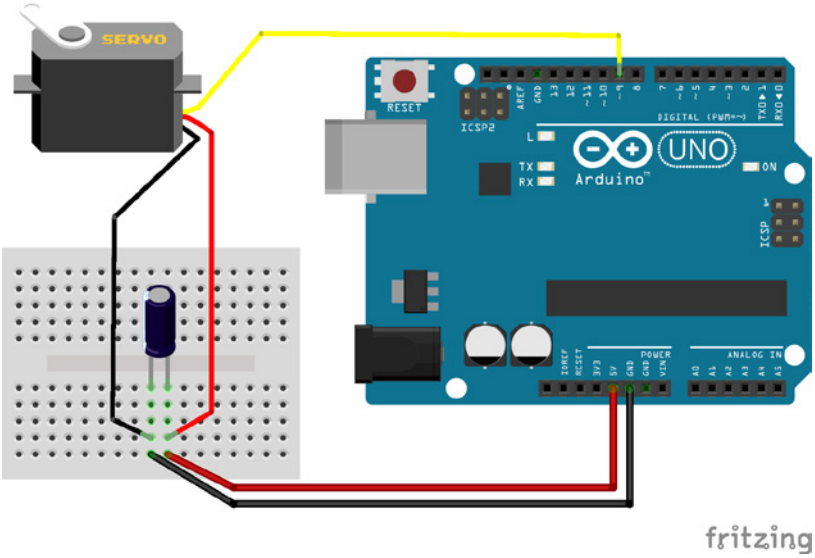
Usando las librerías: servomotor y sensor ultrasónico
Las actividades anteriores nos han permitido adquirir junto a nuestros alumnos importantes competencias en electrónica y programación. Además, nos inspiraron a buscar soluciones a problemas de muy diverso tipo.

Existe una máxima entre los programadores profesionales que fundamenta el uso y proliferación de las librerías en los diferentes lenguajes de programación: "¡No reinventes la rueda!". Es que muchos de los problemas que debemos resolver para llegar a determinada solución ya han sido resueltos por otros desarrolladores. Hacer uso de estas soluciones ya publicadas es una buena práctica, que además de reducir el tiempo y el trabajo que debemos emplear, contribuye al desarrollo de las tecnologías a nivel global.
En este apartado veremos algunos componentes electrónicos muy interesantes para nuestros proyectos de robótica, que podemos aplicar fácilmente usando librerías.

Servomotor
Los servomotores son actuadores que producen un movimiento angular en torno a un eje, al igual que los motores, pero que a diferencia de estos, no giran libremente sino que se colocan en la posición que les indicamos. Algo muy útil para aplicaciones como mover los alerones de un avión, el timón de un barco o el cuello de un humanoide.

Para usarlos debemos conectar tres cables: un polo positivo, uno negativo y uno de control. Se controlan usando escritura analógica, por lo que debemos conectar la terminal de control a uno de las clavijas PWM de Arduino.

[27] IDE, con la que podemos crear una comunicación serial usando otros pines digitales como puerto. www.keuwl.com



El condensador (100uF) que colocamos en paralelo al servomotor sirve para alimentar el pico de corriente que demanda el actuador al comenzar a moverse. Podríamos causar errores por falta de corriente para alimentar el microprocesador si no usamos este componente. Es importante respetar la polaridad de los condensadores, de lo contrario pueden explotar.

La librería que tiene instalada por defecto la IDE de Arduino para controlar servomotores se llama Servo. Encontramos ejemplos de cómo se usa en Archivo>>Ejemplos>>Servo.

Para usar una librería debemos invocarla:

```
#include <Servo.h>
```

Luego crear un objeto del tipo que provee la librería:

```
Servo nuestroServo;
```

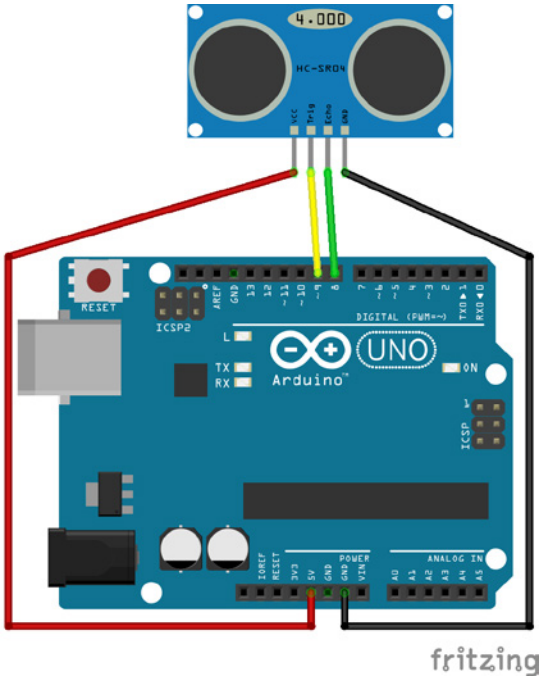
Usamos el método “attach()” del objeto Servo en la función setup() para indicarle al programa dónde lo hemos instalado:

```
void setup(){
  nuestroServo.attach(9);
}
```

Y en el loop() (o en las funciones que invocamos en el loop()) usamos el método write() que espera como parámetro una posición angular en grados (0 a 179 para los servomotores de 180°). Por ejemplo:

```
void loop(){
  nuestroServo.write(45);
  delay(2000);
  nuestroServo.write(135);
  delay(2000);
}
```

Sensor ultrasónico.



El ultrasónico es un tipo de sensor que utiliza el eco que produce un sonido inaudible de muy baja frecuencia para calcular la distancia respecto a una pared u otro objeto contundente. El dispositivo emite una onda sonora cuando excitamos una de sus terminales *(trigger)* y devuelve una señal en otra terminal (echo) cuando la onda retorna al sensor. Calculando el tiempo entre *trigger* y echo y sabiendo la velocidad a la que se desplaza el sonido podemos calcular la distancia que tiene al frente el sensor. Si usamos la librería "Ultrasonic" nos ahorramos esos cálculos y obtenemos el dato en una unidad de medida de longitud, como centímetros o pulgadas.

Esta librería no forma parte del paquete oficial de la IDE de Arduino. Recurriremos a la comunidad para proveernos de una versión actualizada que podemos descargar en: https://github.com/JRodrigoTech/Ultrasonic-HC-SR04/archive/master.zip.

Para instalarla debemos desempaquetar el archivo .zip y copiar la carpeta Ultrasonic con todos sus archivos a la carpeta .../sketchbook/libraries/ que fue creada durante la instalación de la IDE.

Al reiniciar la IDE de Arduino debería aparecer en el menú Archivo>>Ejemplos un elemento nuevo: Ultrasonic, con una lista de sketches de ejemplo que nos muestran cómo usar la librería.
El siguiente sketch de ejemplo (Ultrasonic>>Serial) imprime en el puerto serial, cada décima de segundo, la medida de distancia que devuelve el sensor:

```
// Ultrasonic - Library for HR-SC04 Ultrasonic Ranging Module.
// Rev.4 (06/2012)
// J.Rodrigo ( www.jrodrigo.net )
// more info at www.ardublog.com

#include <Ultrasonic.h>

Ultrasonic ultrasonic(9,8); // (Trig PIN,Echo PIN)

void setup() {
  Serial.begin(9600);
}

void loop()
{
  Serial.print(ultrasonic.Ranging(CM)); // CM or INC
  Serial.println(" cm" );
  delay(100);
}
```

6. Lecciones aprendidas y conclusiones

6.1. Acerca de los procesos de cambio

A nivel global, la expansión de las tecnologías transforma distintas esferas de la sociedad así como diversas actividades de los individuos. Identificar estas transformaciones no alcanza para desplegar los procesos de cambio institucional. Es necesario sortear una serie de desafíos extremadamente complejos, que se deben traducir en objetivos y acciones específicas:

• Conectar las instituciones con lo que pasa en la vida de los NNA , porque los conocimientos y los procesos de aprendizaje comienzan con lo cercano.

• Recrear diseños institucionales de proximidad, reconociendo la integralidad de los estudiantes.

• Incorporar una serie de habilidades y capacidades que nos permitan potenciar los procesos de investigación y construcción de soluciones.

• Utilizar el aprendizaje basado en proyectos implica superar una serie de restricciones vinculadas a los espacios y tiempos de la organización escolar.

• Acoplar las instituciones a los cambios a nivel global, contribuyendo a que NNA accedan a los aprendizajes que no se encuentran extendidos en la oferta de educación básica tanto de Primaria como de Secundaria.

• Incluir el PC supone integrarlo como una dimensión que atraviesa una diversidad de actividades y que por lo tanto trasciende el clásico curso de informática orientado al uso de los programas de computadoras.

6.2. El desarrollo del proyecto Robotic - Pensamiento Computacional

Durante el desarrollo del diseño, planificación e implementación del proyecto Robotic - Pensamiento Computacional, sucedieron una serie de acontecimientos que orientaron al equipo técnico en la toma de decisiones y en las acciones que se fueron realizando.

Estos acontecimientos constituyeron hechos significativos que marcaron momentos importantes en el desarrollo de los procesos, en la vida de los grupos y de cada uno de sus participantes. En algunos casos sirvieron para consolidar acciones que se venían realizando y en otros permitieron ajustar la propuesta.

Reconstruimos la trayectoria recorrida, identificando las fortalezas y valorando qué líneas de acción se deben destacar. Repasamos algunas lecciones aprendidas que entendemos son claves para transitar por la propuesta con grupos de NNA.

- A cada proceso grupal su propia currículo.

Destacamos los diferentes caminos que fueron tomando cada uno de los grupos participantes, reconociendo los intereses y los proyectos que surgían y que definieron la narrativa grupal e individual. Los participantes se apropiaron de la propuesta, investigaron, mejoraron el producto, elaboraron nuevos objetivos y continuaron con el proyecto.

- La alianza con los docentes fue un condicionante en la construcción de proyectos grupales, planificando, implementando y evaluando la propuesta.

- La potencia del juego.

¿Qué sucede cuando se proponen juegos como herramienta de aprendizaje? La estructura del espacio escolar se vio cuestionada. Los niños se sorprendieron de que la actividad propuesta fuera jugar videojuegos, lo que evidencia el lugar secundario que tiene lo lúdico en los currículos de la educación formal. Se vieron cuestionadas las restricciones del marco disciplinario que organiza la convivencia desde la autoridad, devolviendo a los niños la responsabilidad por la organización del espacio, la convivencia y los aprendizajes. Esto implicó reconocer la potencia del juego como herramienta que contribuye a la incorporación de aprendizajes significativos, lo que permitió que los niños movieran sus roles establecidos, tomando contacto con otras dimensiones de su identidad.

- Conocer, conectar, colaborar y proyectar.

Durante el desarrollo de la propuesta y a medida que se fue avanzando en la elaboración de proyectos, se organizaron y coordinaron distintos eventos que contribuyeron a formar parte de una comunidad motivada por una temática común e interesada en compartir sus procesos.

- Las visitas al proyecto de sensibilización en robótica que lleva adelante UyRobot y Sinergia Tech (Centro de desarrollo tecnológico y laboratorio de fabricación digital) contribuyeron a motivar los proyectos grupales, ayudaron a visualizar las distintas etapas para lograr el producto final e impulsaron a varios de los participantes a proyectarse en el mundo de las tecnologías.

- La participación en varios encuentros como el SUMO (encuentro de robótica), las Olimpíadas Ceibal (programación, robótica y videojuegos) y la

Roboteca (muestra de robótica de centros educativos) aportó sentido, motivación y objetivos concretos a los proyectos grupales. El encuentro con otros motivó a los participantes a proyectarse; permitió la circulación social y la integración; favoreció la construcción de comunidad, el intercambio y la educación entre pares; fortaleció lo grupal y la apropiación del producto. Desarrolló habilidades comunicacionales necesarias para relatar y describir el proceso que llevó a la construcción del producto final.

• La organización de la 1era. Feria de Pensamiento Computacional, en la que todos los grupos presentaron los productos de los proyectos elaborados, posibilitó compartir el trabajo realizado durante el año, así como hacer visible la temática y difundir el proyecto.

• Actividades puntuales de difusión y acercamiento a la temática. La organización de la Primavera Computacional -que implicó una serie de encuentros puntuales en distintos centros- nos permitió sensibilizar en la temática, difundir y generar las condiciones para que más actores del ámbito educativo se animen a utilizar estas herramientas. Las experiencias más exitosas se llevaron a cabo en los centros educativos que por un lado demostraron interés en la temática y que además contaban con kits de robótica, pero planteaban la necesidad de formación para su utilización.

6.3. A modo de cierre

Incluir la temática de las tecnologías y específicamente el Pensamiento Computacional en el campo de la educación es un desafío que afrontamos juntos Gurises Unidos con Fundación Telefónica - Movistar. La experiencia adquirida durante estos años de trabajo colaborativo con centros educativos, buscando mejorar el acceso y la calidad de la educación, nos permitió desarrollar una visión situada en las dificultades y potencialidades para la implementación de propuestas innovadoras. A partir de la combinación de saberes en el área de la educación, del trabajo con los NNA y del campo de las tecnologías logramos co-crear la primera propuesta de Pensamiento Computacional para el ámbito educativo en Uruguay.

Incorporar las tecnologías nos obligó a comenzar un proceso de investigación y formación específico, que además implicó el esfuerzo por conectar esta temática con el conjunto de dificultades que atraviesa la educación, en especial a la que acceden los NNA. Los resultados de este esfuerzo se aprecian en la disposición a los aprendizajes, la motivación y las habilidades que despliegan los NNA en cada una de las propuestas educativas planteadas.

Durante el desarrollo de este proyecto han participado de manera sistemática más de 10 centros educativos de primaria, educación media y educación no-formal. Además, muchos otros centros educativos públicos y privados han demostrado su

interés por llevar adelante propuestas en esta línea. Es posible que otros centros educativos repliquen las propuestas, incorporando nuevos elementos, compartiendo y multiplicando los proyectos que surjan de estas experiencias. Recientemente hemos puesto en marcha, en alianza con el Departamento de Tecnología de Primaria de la ANEP, un curso de formación dirigido a docentes de todo el país, con el objetivo de incorporar de manera transversal el PC en las prácticas educativas. A través de una formación semipresencial, los docentes adquieren conocimientos en la temática utilizando la metodología de aprendizaje por proyectos.

Por último, pensar un área nueva con escasa acumulación previa nos recordó algo extremadamente importante: en el vínculo educativo, una actitud docente de enseñar aprendiendo y aprender enseñando es clave para despertar en otros la idea de que todos y todas podemos aprender y que para ello es necesario tener un rol activo.

Bibliografía

AGESIC. Encuesta Específica de Acceso y Uso de TIC (EUTIC 2016). En: https://medios.presidencia.gub.uy/tav_portal/2017/noticias/NO_X086/2017-05-17%20EUTIC2016.pdf

ASAMBLEA GENERAL DE LAS NACIONES UNIDAS (1948). Declaración Universal de Derechos Humanos.

ASAMBLEA GENERAL DE LAS NACIONES UNIDAS (1989). Convención sobre los derechos del niño. En: https://old.unicef.es/sites/www.unicef.es/files/CDN_06.pdf

BLANCO, Rosa (Responsable general) (2007). Educación de calidad para todos: un asunto de derechos humanos. Documento de discusión sobre políticas educativas en el marco de la II Reunión Intergubernamental del Proyecto Regional de Educación para América Latina y el Caribe. EPT/PRELAC - UNESCO Oficina Regional de Educación para América Latina y El Caribe, Santiago. http://unesdoc.unesco.org/images/0015/001502/150272s.pdf

BRAVO-LILLO, Cristian (2015). "Pensamiento Computacional: una idea a la que le llegó el momento." En revista Bits de ciencia N° 12 (pág. 48-51). Departamento de Ciencias de la Computación de la Universidad de Chile. https://www.dcc.uchile.cl/Bitsdeciencia12.pdf

CSTA e ISTE (2011). Pensamiento Computacional. Caja de herramientas para líderes. (Primera Edición). http://eduteka.icesi.edu.co/pdfdir/PensamientoComputacional1.pdf

ENAJ, 2013. INJU. Tercera Encuesta Nacional de Adolescencia y Juventud.

FERRY, Gilles et al. (1976). Nuevas actitudes en la relación pedagógica. Librería del Colegio, Buenos Aires.

FITZGERALD, Scott y SHILOH, Michael (2014). El libro de proyectos arduino. Arduino srl, Turín.

HITSCHFELD, Nancy et al. (2015). "Pensamiento Computacional y programación a nivel escolar en Chile: El valor de formar a los innovadores tecnológicos del futuro." En revista Bits de ciencia N° 12 (pág. 28-33). Departamento de Ciencias de la Computación de la Universidad de Chile. https://www.dcc.uchile.cl/Bitsdeciencia12.pdf

KEMPA, R.F. (1986) “Resolución de problemas de Química y estructura cognoscitiva.” En revista Enseñanza de las Ciencias, 4 (2), 99-110.

MUÑOZ, Vernor. “El derecho a la educación: una mirada comparativa Argentina, Uruguay, Chile y Finlandia.” UNESCO, s/f. En: http://portal.unesco.org/geography/es/files/15017/13230888961Estudio-comparativo-UNESCO-Vernor-Munoz.pdf/Estudio-comparativo-UNESCO-Vernor-Munoz.pdf

OECD. “¿Los jóvenes de 15 años son creativos a la hora de resolver problemas?” PISA IN FOCUS 38. Abril 2014. En: http://www.mecd.gob.es/dctm/inee/pisa-in-focus/pisa-in-focus-n38-esp-1-4-2014.pdf?documentId=0901e72b81904916

PODER LEGISLATIVO DE LA REPÚBLICA ORIENTAL DEL URUGUAY (2008) Ley General de Educación. IMPO - Montevideo. Disponible en: http://www.impo.com.uy/bases/leyes/18437-2008

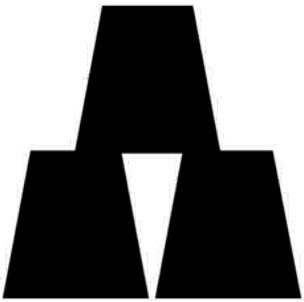
PRENSKY, Marc. “Nativos e Inmigrantes Digitales.” En Cuadernos SEK 2.0.

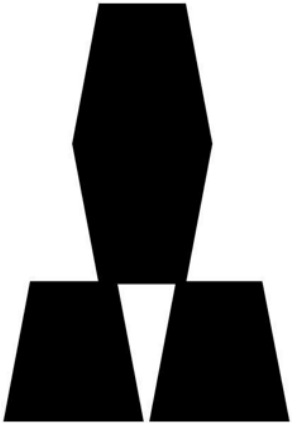
REYES, Reina (1964) El derecho a educar y el derecho a la educación. Editorial alfa. Montevideo.

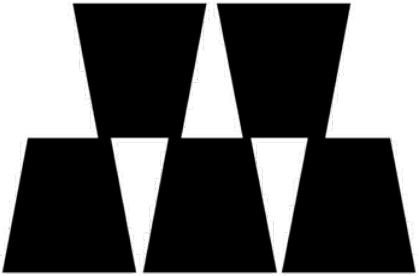
VALVERDE BERROCOSO, Jesús. “El pensamiento computacional y las nuevas ecologías del aprendizaje”. En: http://www.um.es/ead/red/46/valverde_et_al.pdf

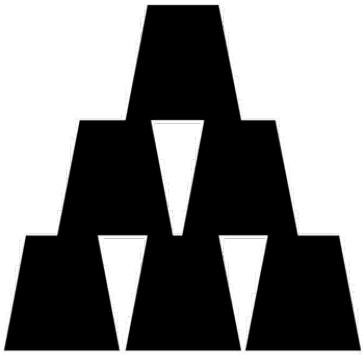
WING, Jeannette. (2006) “Computational Thinking”. En: https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf

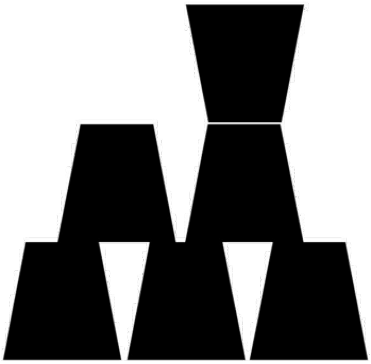
Anexo 1: Desafíos

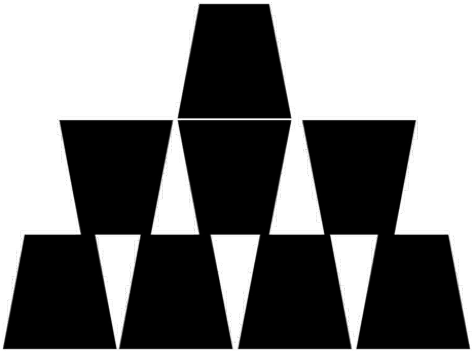


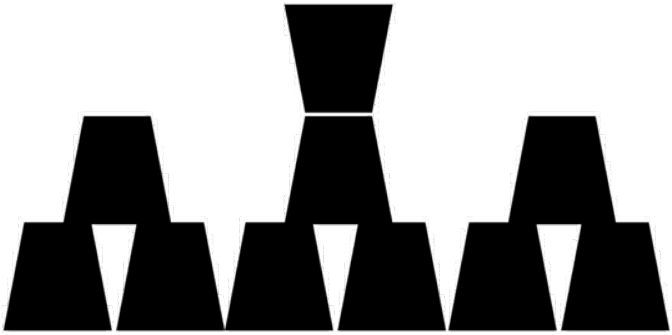


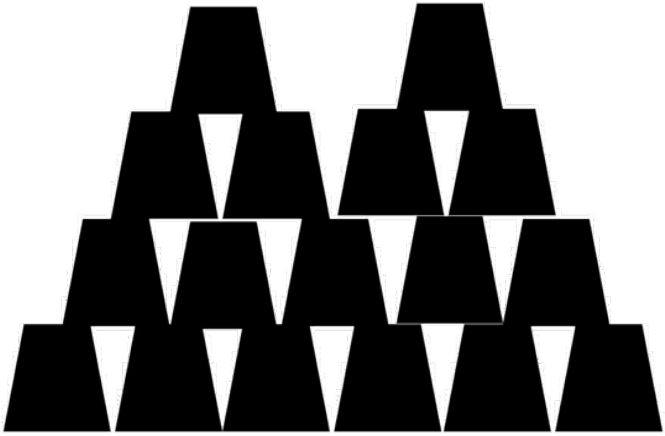










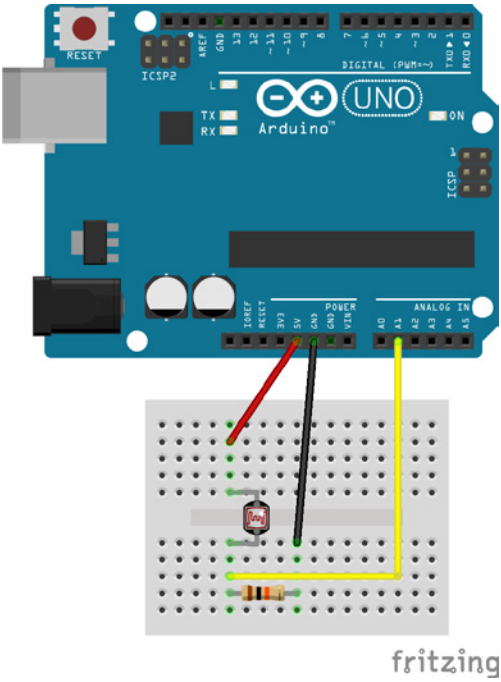


Anexo 2: Construyendo sensores analógicos

Nombre actividad	Construyendo sensores analógicos
Objetivo	Construir sensores analógicos que nos permitan medir diferentes magnitudes físicas usando componentes discretos (resistencias y transistores)
Áreas de conocimiento	Electrónica. Física. Programación
Conceptos relacionados	Fotorresitencia. Sensor resistivo. Fototransistor. Termistor. Resistencia pull-down. Habilidades asociadas
Materiales	Para cada subgrupo: Computadora con IDE instalado Placa Arduino Cable USB Protoboard Cables puente para protoboard Fotorresistencias (u otro tipo de sensor resistivo) Fototransistor LED infrarrojo 1 resistencia 10kΩ 1 resistencia 220Ω 1 resistencia 4.7kΩ
Enlaces de interés	

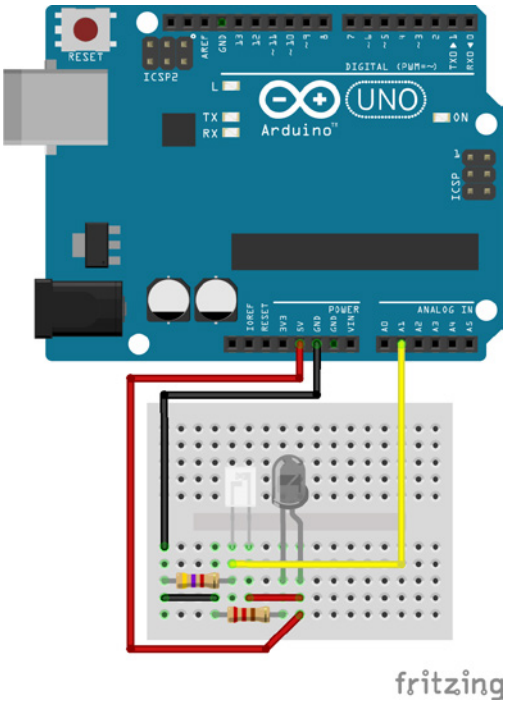
Además de los potenciómetros, que como sensores nos permiten medir el estado de un elemento físico en función del ángulo de giro en torno a un eje, existen varios componentes de muy bajo costo que podemos conectar a las entradas analógicas de Arduino para medir diferentes cosas:

Podemos usar fotorresistencias para medir el nivel de luz. Las resistencias (incluidas las fotorresistencias) no tienen polaridad, por lo que simplemente conectamos el polo positivo (5v) a un extremo y al otro extremo la tierra (mediante una resistencia "pull-down" de 10kΩ) y a la entrada analógica de Arduino. Así conectamos una fotorresistencia al pin analógico 1:



Del mismo modo podemos conectar otros sensores resistivos como los que miden presión, curvatura o temperatura.

Podemos construir otro tipo de sensor analógico usando termistores y fototransistores. En el siguiente esquema conectamos un fototransistor infrarrojo lado a lado con un LED infrarrojo. De esta manera podemos medir la cantidad de luz infrarroja que se refleja en una superficie. Un sensor muy útil para medir la claridad (en escala de grises) de una superficie, o la distancia respecto a una superficie de color constante:



A simple vista los fototransistores se parecen mucho a los LED y son también componentes polarizados. Tienen tres terminales como otros transistores, solo que no vemos la compuerta (la que controla el flujo), que se encuentra internamente conectada a un material fotosensible. Y atención, porque la fuente (o colector), que es donde debemos conectar el polo positivo (5v) es la pata más corta, al contrario que en los LED. La pata larga (drenaje o emisor) se conecta a la tierra (mediante una resistencia "pull-down" de 4,7kΩ) y a una entrada analógica de Arduino.
Pongamos a prueba estos sensores usando el sketch de ejemplo AnalogReadSerial. ¿Cuántos divertidos proyectos se nos pueden ocurrir usando algunos actuadores y estos sensores analógicos?

GLOSARIO

Algoritmo: *Es una lista de instrucciones claras, ordenadas y precisas que describen un procedimiento para la solución de un problema.*

Archivo Binario: *Es un archivo informático que contiene información de cualquier tipo codificada en binario para el propósito de almacenamiento y procesamiento en ordenadores. Muchos formatos binarios contienen partes que pueden ser interpretadas como texto. Un archivo binario que solo contiene información de tipo textual sin información sobre el formato de él se dice que es un archivo de texto plano. Habitualmente se contraponen los términos 'archivo binario' y 'archivo de texto', de forma que los primeros no contienen solamente texto.*

Código abierto: *El software de código abierto es un software que puede ser libremente usado, cambiado y compartido (en forma modificada o no modificada) por cualquier persona. El software de código abierto es hecho por muchas personas y distribuido bajo licencias que cumplen con la Definición de Open Source .*

Extraído de https://opensource.org/

Depurar: Es el proceso por el cual un desarrollador revisa un programa corrigiendo los errores y mejorando el código.

Estructuras de control: *Son una serie de instrucciones que se pueden usar en un lenguaje de programación para controlar el flujo de ejecución de un programa, es decir, controlar las condiciones bajo las que se deben ejecutar las diferentes partes del código.*

Hardware: *Conjunto de los componentes que integran la parte material de una computadora u ordenador.*

Infotecnologías: se refiere a cualquier tecnología que ayuda a producir, manipular, almacenar, comunicar, difundir información.

Librerías: *Conjunto de funciones para la solución de problemas específicos, organizada como una caja de herramientas, creadas en un cierto lenguaje de programación. Se utilizan para simplificar o ampliar y mejorar la creación de nuevos programas o funciones.*

Extraído de http://lema.rae.es/dpd/srv/search?key=hardware

Pseudolenguaje o Pseudocódigo: *Es un lenguaje artificial e informal que auxilia a los programadores a desarrollar los algoritmos. Este es similar al lenguaje natural, es amigable, aunque no se trate de un lenguaje verdadero de programación de computadoras. Es un lenguaje intermedio entre el lenguaje natural y los lenguajes de programación.*

Extraído de https://www.fing.edu.uy/tecnoinf/mvd/cursos/prinprog/material/teo/prinprog-teorico01.pdf

Software: *Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.*

Extraído de http://dle.rae.es/srv/search?m=30&w=software

Tipo de datos: *El tipo de dato informático es un atributo de una parte de los datos que indica al ordenador (y/o al programador) algo sobre la clase de datos sobre los que se va a procesar. Esto incluye imponer restricciones en los datos, como qué valores pueden tomar y qué operaciones se pueden realizar. Tipos de datos comunes son: enteros, cadenas alfanuméricas, fechas, horas, colores, coches o cualquier cosa que se nos ocurra. Por ejemplo, el tipo "int" representa un conjunto de enteros. Este es un concepto propio de la informática, más específicamente de los lenguajes de programación, aunque también se encuentra relacionado con nociones similares de las matemáticas y la lógica.*

Extraído de https://www.fing.edu.uy/inco/cursos/fpr/wiki/index.php/Variables_y_Tipos

ToolKit: *Caja de herramientas para líderes de Pensamiento Computacional creado por la Asociación de Docentes en Ciencias de la Computación (CSTA)*

Extraído de: http://eduteka.icesi.edu.co/pdfdir/PensamientoComputacional1.pdf

Variables: *En programación, las variables son espacios reservados en la memoria que, como su nombre indica, pueden cambiar de contenido a lo largo de la ejecución de un programa.*

Extraído de https://www.fing.edu.uy/inco/cursos/fpr/wiki/index.php/Variables_y_Tipos

El Pensamiento Computacional apunta a generar una forma de pensar donde se aprende a plantearse problemas y sus soluciones, cumpliendo una secuencia determinada de pasos en el proceso.

"Pensamiento Computacional: un aporte para la educación" es una publicación que ayuda al docente a abordar esta propuesta en el aula desde una forma práctica. Es el resultado de experticias en el área educativa, en el trabajo con los niños, niñas y adolescentes en el área de las tecnologías.

www.fundaciontelefonica.uy

Fundación Telefónica Uruguay
@Ftelefonicauy
Fundación Telefónica Uruguay

Telefónica FUNDACIÓN / movistar